SYSTEM CONFIGURATION FOR PROPORTIONAL CONTROL OF AN ASSISTIVE TECHNOLOGY FOR PATIENTS WITH CERVICAL SPINAL CORD INJURIES

A Dissertation
Presented to
The Academic Faculty

by

Nathan VerDon Parrish

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
December 2018

SYSTEM CONFIGURATION FOR PROPORTIONAL CONTROL OF AN ASSISTIVE TECHNOLOGY FOR PATIENTS WITH CERVICAL SPINAL CORD INJURIES

Approved by:

Professor David V. Anderson, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Associate Professor Omer Inan
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Mark Davenport
School of Electrical and Computer Engineering
Georgia Institute of Technology

Assistant Professor Joshua Kacher
School of Materials Science and Engineering
Georgia Institute of Technology

Associate Chair Benjamin Klein
School of Electrical and Computer Engineering
Georgia Institute of Technology

Date Approved: Nov 5, 2018

To my wife, Elizabeth. Without her constant support, encouragement, and patience, I would have never been able to complete this journey. Also to Gabriel, Katarina, Alexander, and Elijah who joined us along the way.

PREFACE

This work began as an exploration to improve upon past research of an assistive technology for people with cervical spinal cord injuries. Using a magnet placed on the tongue and with a headset containing magnetic sensors, a person could control a computer, smart phone, or wheelchair with classification. This work began as the search for the ability to provide more fluid control by proportional tracking of the magnet. In the process of trying to accurately characterize the model and its parameters, it was discovered that performing the model fusion and characterizing the model framework for performing the tracking was itself an interesting set of problems and became the focus of the work. Hopefully the techniques identified here will prove to be useful tools for enabling data fusion with an array of sensors measuring common values for characterizing a system within the framework of a model.

ACKNOWLEDGEMENTS

I want to thank my advisor, Dr. David Anderson, for all his assistance in this work. He was always a faithful advocate, a source of sound advice, an example of strict integrity, and a model of finding a balance between personal and professional life. He has been a singularly exceptional advisor, a valued mentor, and a faithful friend.

I would also like to thank Dr. Maysam Ghovanloo who introduced me to the problem and who provided me the opportunity to work with his company to develop practical elements and commercialization of the device that served as the center of my research.

I appreciate the help of Md. Nazmus Sahadat who provided me information and assistance pertaining to the hardware considerations of the device and who helped me to stay abreast of the work at the GTBionics Lab.

Appreciation to the members of the Efficient Signal Processing Lab who made themselves available to discuss the problem and who served as a sounding board for investigating this work.

I must also acknowledge the efforts of my parents, VerDon and JoAnn Parrish, for providing me with the experiences and opportunities that have allowed me to come to this point in my life. They provided me the example of how to be a good person, instilled in me a love of learning, and provided constant encouragement.

Appreciation to my in-laws, Dr. Blaine and Nancy Hart. Their assistance and support, especially regarding my family, gave me the flexibility to complete this journey. I also wish to thank Blaine Hart for his review of the medical elements of the thesis and, furthermore, for his being the first person to read my thesis in its entirety. His notes and corrections are greatly appreciated and served to improve the quality

of the overall work.

Finally, I would like to thank the members of my committee. Thanks to Dr. Mark Davenport, Dr. Omer Inan, Dr. Benjamin Klein, and Dr. Joshua Kacher for their participation in this process.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Modern technology has lead to major changes in how people interact with the world. Not so long ago, phones were limited nearly exclusively to landline connections and computers were limited to business tasks. Now smartphones are becoming ubiquitous and small embedded computers are found in an ever increasing collection of devices comprising the Internet of things.

This technological development provides new opportunities for people with disabilities. The connectivity of an increasingly broad set of devices provides a pathway to allow people with disabilities to have far greater opportunities to interact with their world, if the proper devices can be made to connect them to that infrastructure.

To this end, work has been done at Georgia Tech to create an assistive technology device for people with tetraplegia. The assistive technology is a wearable headset with arms that come down both sides containing small magnetic sensors. These sensors track the position of a rare-earth magnet that is placed on the user's tongue. This magnet can be temporarily placed using a dental adhesive, or it can be placed more permanently by a piercing or embedding the magnet within the tissue of the tongue. This allows the person to, by means of tongue movement, issue commands to a computer or smartphone device.

There has been a significant amount of work done in developing this system at the GTBionics lab at Georgia Tech. Past research has largely focused on performing classification tasks, creating a dependable communication framework, and making progress towards a multimodal system. There has also been some effort to try and perform more sophisticated tracking of the magnet, but so far has proven to be

unsuccessful.

There are a few major problems to correctly being able to track the magnet position. First, the magnetic dipole is a nonlinear system. There is also the consideration of various mismatch of the sensors in position and orientation. The tracking itself can be performed by using established filtering techniques such as particle filters, extended Kalman filters, or unscented Kalman filters, but this requires a thorough characterization of the system to provide accurate results.

The goal of the research described herein is to address these underlying challenges to create the underlying framework to allow successful tracking algorithms to be performed.

CHAPTER II

MEDICAL BACKGROUND AND ASSITIVE TECHNOLOGIES

To better understand the goals of the project, an overview of spinal cord injuries is presented with a focus on cervical spinal cord injuries. A description of some of the available assistive technologies is presented. Some existing technologies that are similar to the proposed work are then described.

2.1 Spinal Cord Injuries

The level of remaining capability for people with spinal cord damage can typically be classified based on the location and nature of the injury. Nerve connections are made with the spinal cord between pairs vertebrae. Each vertebra is categorized into one of three categories: the cervical, the thoracic, and the lumbar. These regions consist of seven, twelve, and five individual vertebrae, respectively. The cervical nerves are, therefore, labeled C1 to C8. C8 is just above the vertebra T1, and the nerve T1 is just below the first thoracic vertebra. To help distinguish between the vertebra and the nerves, the nerves will be denoted as CN1-CN8. People with cervical spinal cord injuries lack full control of their arms and legs, and they are typically said to have tetraplegia.

In addition to the location of the injury, the nature of the injury can also have a large inpact. The injury can be asymmetric which can lead to differences in capabilities from one side of the body to the other. For example, a person may be able to move their head more freely to the right than to the left. An approximate representation of the approximate muscle groups controlled by each spinal nerve is

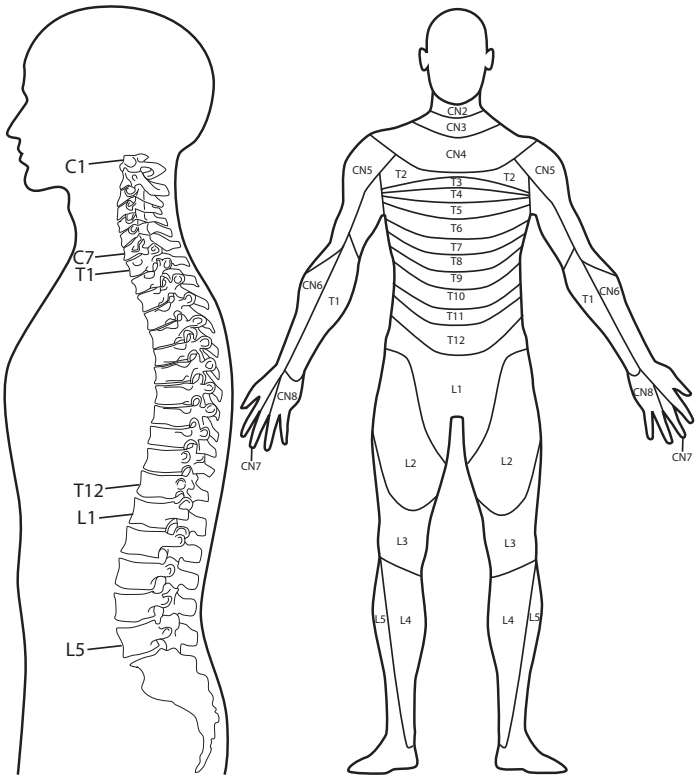


Figure 1. In the image on the left, the terminal vertebrae in the cervical (C), thoracic (T), and lumbar (L) regions are identified with the enumeration of the intermediate vertebrae being implied. The spinal nerves correspond to these vertebrae and, on the right, the muscular regions of the body are approximately identified by their corresponding spinal nerves.

depicted in Figure 1.

Individuals with injuries between CN1 and CN3 lack neck muscle control and innervation t othe diaphragm and likely depend upon a ventilator for breathing. With injuries at CN4, the individual retains some neck control. CN5 injuries may permit elbow flexion and CN6 permits wrist extension. CN7 injuries may allow the patient to extend their elbow. Generally, CN7 is the highest level of injurty where a person is capable of independent living. Injeries at or below CN8 preserve some level of finger flexion, but the patient has significantly limited dexterity.

2.2 Existing Pertinent Assistive Technologies

A variety of assistive technologies for individuals with tetraplegia are depicted in Figure 2.

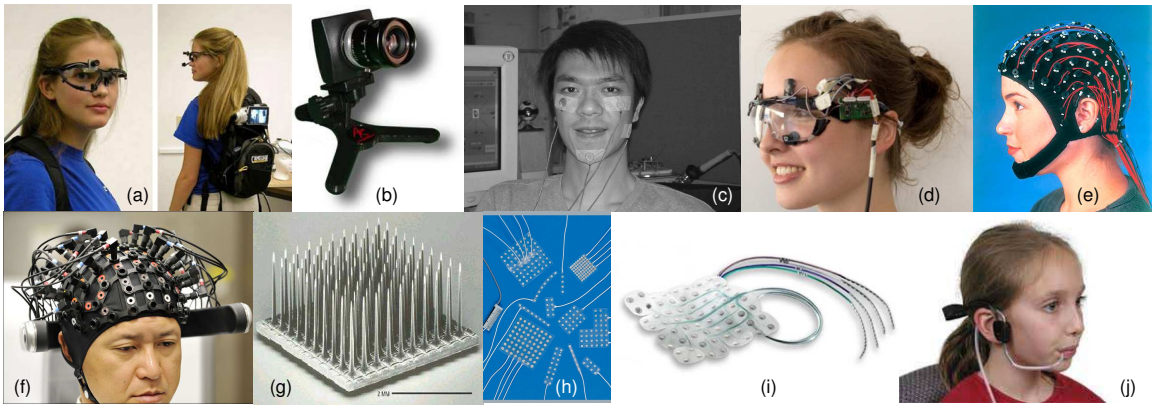


Figure 2. Examples of existing ATs for people with tetraplegia: (a) Wearable eye-tracker system [9], (b) Eye-gaze camera [2], (c) Facial electromyograph [32], (d) Wearable electrooculography system [13], (e) Electroencephalography headset [72], (f) Near-infrared-based electroencephalography headset [50], (g) Intracortical electrode array [38], (h) Subdural strip electrode array [4], (i) Another subdural strip electrode array [3], (j) and Sip-and-puff system [1]

Typically, able-bodied users use their hands and fingers for human-computer interaction, whether using a keyboard or a mouse. These input methods have a high bandwidth when compared with most assistive technologies. People with high-level spinal cord injuries must use alternative input methods. The preferred assistive technology (AT) devices vary by level of injury. The currently available devices are listed from the highest, most-severe injuries to the lowest, least-severe injuries.

2.2.1 CN1 to CN3

For injuries between CN1 and CN3, available assistive technologies must rely almost exclusively on the voluntary musculature of the face and head. HCI devices based on eye-tracking have been successfully employed [9, 10], and have also been used for individuals with amyotrophic lateral sclerosis (ALS), a neurodegenerative disease.

Eye trackers are effective since eye control has one of the most direct connections to the brain. Eye trackers, however, can interfere with normal visual tasks and the input devices are sensitive to ambient conditions (e.g., lighting.) Also, like many devices herein, they are vulnerable to the "Midas touch" problem, a term used to describe when the input system accepts inputs from user actions which were not intended to be used for information entry. With eye tracking, this can be from the user observing the room or eye saccades, natural movement of the eyes as they scan the visual scene. Techniques have developed to deal with some of these problems and eye tracking can be faster than mouse selection [57], but at the expense of fine precision.

Another option for people with injuries from CN1 to CN3 is electromyography (EMG), which senses bioelectric signals from muscles in the face as a form of input [32]. A related technology is electrooculography (EOG), which senses the electric potential field generated by the positively charged cornea with the negatively charged retina [65]. EMG requires electrodes to be placed on the skin. EOG can be implemented by sensors on glasses [13]. In either case, these technologies have bandwidth limitations and are vulnerable to noise. The Midas touch problem is also an issue. Facial EMG has also been used for identifying emotional response in psychology research.

Other devices aim to directly interface with the brain. These brain-computer interfaces (BCI) come in two varieties: invasive and noninvasive. Noninvasive BCI use either electroencephalography (EEG) [72] or near infra-red (NIR) signals [50]. EEG devices have low data-input rates with recent research reporting maximum bit-rates about 100 bits/min in a typing task [69]. Near infra-red BCI are still immature, so data rates are still low. Invasive BCIs use intracortical neural signals [38] or subdural electrocorticogram (ECoG) [41]. These devices are costly and require sensors to be placed in direct contact with the brain. Movement of the brain within the skull is a major obstacle - one that ECoG attempts to address by using surface contacts that

allow some movement without significant shifting. Furthermore, BCIs must be trained on an individual basis and are vulnerable to a variety of noise sources. Nevertheless, accuracy of an intracortical BCI demonstrated an acceptable level of accuracy more than 1000 days after the device had been placed [58].

2.2.2 CN4

With injuries at CN4, the individual has some neck control. At this point, head motion becomes a possible input method.

Diaphragm control allows the use of sip-and-puff devices. Sip-and-puff devices have been used for a long time for controlling wheelchairs. Sip-and-puff continues to be widely used, e.g., a wireless remote control for an Apple iPod was modified to use sip-and-puff to help a user with cervical spinal cord injury control their music independently[36].

Voice is also an available mode for individuals with diaphragm control. One study demonstrated a vocal joystick that made use of vowel quality, pitch, and loudness features to control a proportional input system [11]. While speech can be used by people with higher injuries with the assistance of a ventilator, the inability to control their diaphragm limits some ATs for injuries above CN4.

2.2.3 CN5-CN8

Below C7 injuries, a certain level of input using the hands becomes possible, for example using a hand control or joystick to control a wheelchair. Typing and the fine use of a computer mouse remain difficult or impossible. Devices do exist that modify keyboards or mice to improve usability for people with various disabilities.

2.3 Challenges of Assistive Technologies

Fatigue is a common problem with AT [16]. Allowing multiple modes allows for a user to change his/her preferred mode as they become fatigued (e.g., use a less-efficient but less-strenuous method). Consideration of user mental workload is also important. Novak, Mihelj, and Munih showed that when users were asked to perform two tasks concurrently and observed the psychophysiological effects. They showed that users could become mentally fatigued as they reached the limits of their mental resources as they tried to split their attention between the two tasks [46]. A poorly-implemented multimodal HCI could potentially act as a dual-task problem which would run contrary to the motivation for using a multimodal system.

Abandonment is a major obstacle for successful ATs. Research by Phillips and Zhao found four factors associated with AT abandonment: lack of consideration of user opinion, easy device procurement, poor device performance, and change in user needs or priorities [48]. Each of these should be considered in an AT system. One interesting result was that when an AT was easy to procure, there was an increased rate of abandonment because the ease of acquisition reduces the apparent value of the AT to the user. The proposed system requires the user to have a magnetic tracer placed and the system would need to be adapted for an individual user's needs. This customization is necessary to the success of the design, but customization should also serve to reduce abandonment.

The Midas Touch problem is also a major consideration, where commands are issued involuntarily when the user is engaged in unrelated activities.

2.4 Similar Existing Assistive Technologies

Another possible input is to use the tongue for input. There are a number of existing HCIs based on voluntary oral movements [25]. Some depictions of existing technology are represented in Figure 3. The Tongue-Touch-Keypad (TTK) is a switch-based

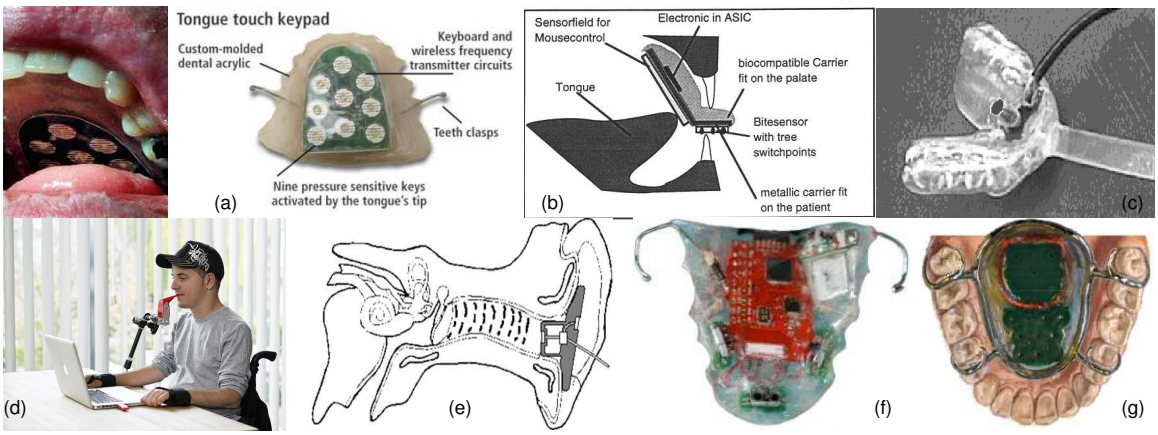


Figure 3. Examples of existing tongue-based ATs: (a) Tongue Touch Keypad (TTK) [7], (b) Tongue-Mouse [47], (c) Tongue-Point [54], (d) Integra-Mouse [5], (e) Think-A-Move [66], (f) Optical tongue gesture detector [55], (g) Inductive tongue-computer interface [63]

device with nine keys that can be activated by tongue pressure [7]. Tongue mouse is a touch-pad which is placed in the mouth and is controlled by using the tongue [47]. Tongue-Point [54] and Integra-Mouse [5] are modified joysticks specialized to be used by tongue and lips. Think-A-Move measures pressure changes in the ear canal as a result of tongue movements [66]. There is also a solution which uses infrared optical sensors embedded within a retainer to sense tongue gestures [55]. Building upon the idea of the TTK system, another project used a metallic activation unit, placed on the tongue, to interact with inductive sensors in an intraoral controller with 18 inductive switches [63]. Using a metallic tracer helps avoid unintentional commands. All of these ATs require bulky devices to be placed in the mouth, interfering with eating and speech.

CHAPTER III

SYSTEM ARCHITECTURE

This chapter describes the architecture of the prototype system and some of the technical limitations of the pertinent elements.

3.1 Overview

The system developed to analyze this system is based on work performed by Bionic Sciences, Inc. as a part of an NSF-SBIR grant #1621673, SBIR Phase I: Developing the Standalone Tongue Drive System. The author was involved with the development with the software and firmware of this system which was performed under contract. The software and firmware developed in the work related to this SBIR is distinct and separate from the work presented as part of this thesis, however the development of the hardware and some aspects of the firmware are pertinent as providing the framework for the algorithms and methods described in this document. This chapter focuses on the hardware development efforts done as part of the SBIR; my contributions will be noted as it demonstrates experience with working with hardware and speaks to the ability to successfully implement the algorithms described in a working system.

Depictions of a prototype system are included in Figure 4. The headset consists of a microcontroller and a custom PCB board on the top with sensor boards on the two arms, each with two magnetic sensors. Closeups of the microcontroller and the custom board are included.

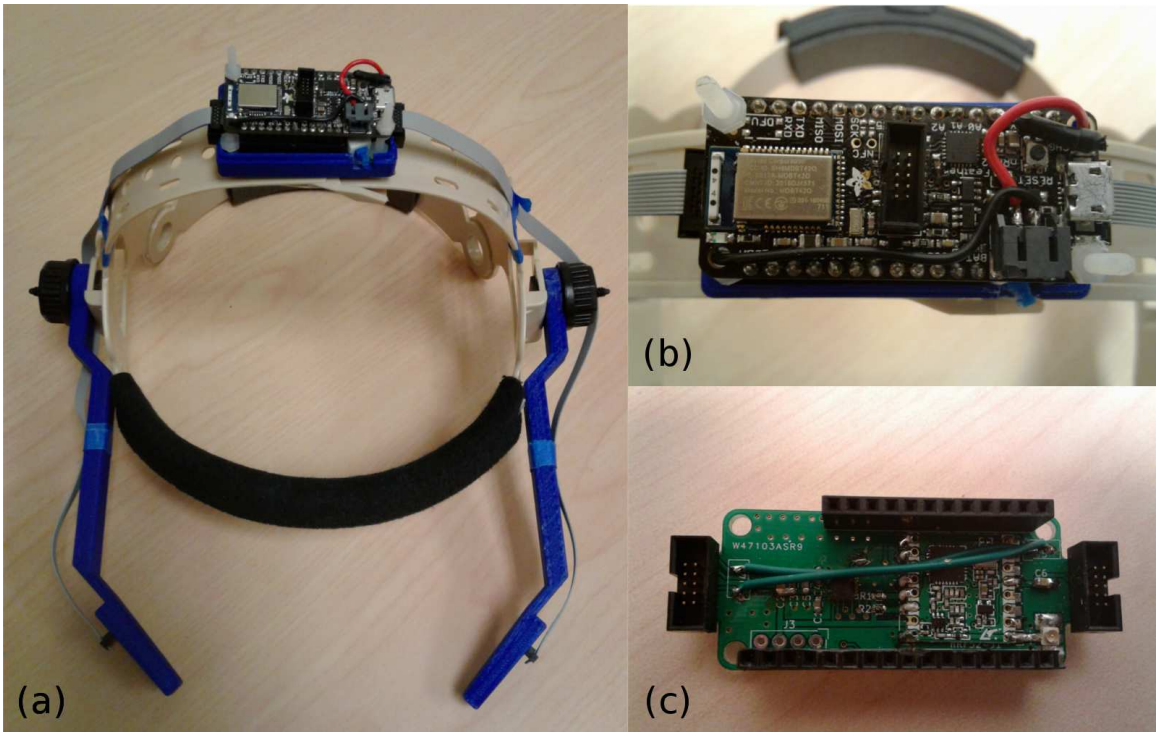


Figure 4. Images of a prototype headset system (a) the complete headset system (b) close-up of the microcontroller (c) close-up of the custom PCB

3.2 Microcontroller

The microcrontroller used in this project is a low-powered processor based on the Nordic nRF52832 system on a chip (SoC) [45]. This SoC contains an ARM Cortex-M4 32-bit floating-point processor running at 64 MHz. It supports a supply voltage from 1.7-3.6V with currents of 1.9μA at 3V in system ON mode. It has a 512 kB flash with 64 kB of RAM. Additionally, it contains support for serial peripheral interface (SPI) and I^2C (Inter-Integrated Circuit) busses with three bus lines available.

Additionally, the microcontroller provides a 2.4 GHz transceiver for implementation of a Bluetooth low-energy (BLE) communication protocol. When communicating, the chip draws 5.4 mA peak current.

There is also support for integrating USB (Universal Serial Bus) communication over a ttyACM (teletypewriter Abstract Control Model) serial connection. A later

version of the chip, the nRF52840, also supports general-purpose USB communication. The nRF52840 microcontroller has not been used in a headset yet, though it has been used as a dongle to communicate with headset systems.

During the development of the hardware system, custom boards were fabricated. My contribution to this effort was evaluating the boards after they had been populated and assessing and solving technical problems as they presented themselves. Later prototypes made use of a Bluefruit nRF52 Feather board. This includes a Bluetooth antenna as well as interfaces providing simple connection points for connecting additional components. It also includes a connection for attaching a rechargeable battery with a power connection to an integrated micro-USB connection. The pinout is depicted in Image 5.

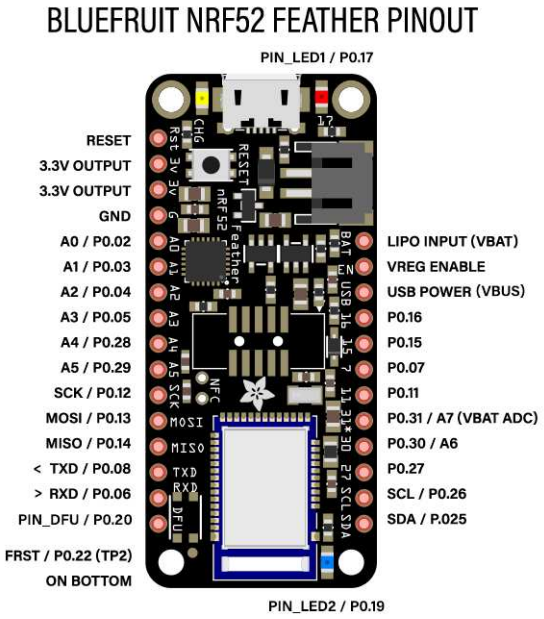


Figure 5. Diagram of the microcontroller system on a chip (SoC) and its pinout [8].

3.3 Architecture

To integrate the feather board with the additional components, a custom hardware board was developed to house some external components as well as to provide connections for communicating with the magnetic sensors placed alongside the user's mouth. This board was connected by pin connections that interface with the connections on the feather board. I contributed to this effort in ensuring that the connections were accurately placed from the standpoint of the microcontroller and testing and debugging the populated interface boards.

The custom PCB board integrates a radio operating in the 433 MHz band as an alternative when the 2.4 GHz bandwidth used by BLE is congested. The chip used for this communication is a Hoperf Electronic RFM69HW transceiver module. This radio transceiver is connected with an SPI connection to one of the SPI busses. This secondary radio is not of primary concern as it pertains to the work presented herein, but it is pertinent to the proper operation of the headset providing additional precautions to ensure connectivity when being used for critical operations such as driving a wheelchair.

There is also a STMicroelectronics LSM9DS1 inertial module that provides a 3D magnetometer, 3D accelerometer, and 3D gyroscope. This is mounted on the PCB and is connected to the Feather board on pins that implement an SPI bus. This SPI bus is shared with all other magnetic sensors since they all share the same SPI communication configuration: the clock is active low with read/write occurring on the falling edge.

To provide communication with a computer or cell phone device, a dongle was created based on an nRF52840 device. The dongle includes a RFM69HW radio transceiver. The current prototype is built with a development board which includes the Bluetooth antennas and mounts for connecting jumper wires to external components, though future designs would be built using a small SoC such as the Rigado

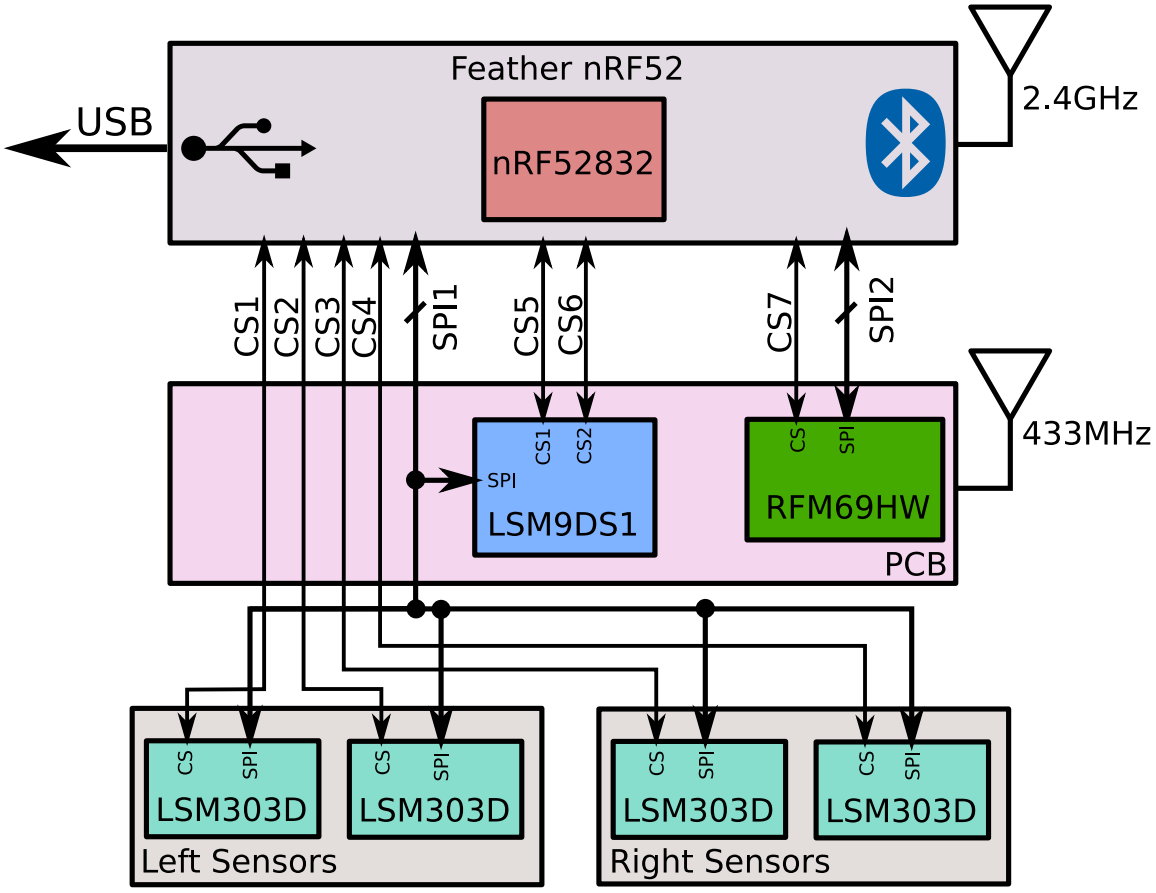


Figure 6. Diagram outlining the system for the headset. Power and ground connections are omitted. There are two SPI busses used and chip select lines are identified.

BMD-340 [34]. The dongle communicates over a USB connection with USB HID and ttyACM communication protocols.

3.4 Magnetic Sensors

There are two magnetic sensor types used in this design. The magnetic sensor that is connected to the custom PCB board is an LSM9DS1 [61]. The LSM9DS1 has two chip select paths. The first chip select communicates with the accelerometer and gyroscope and the second chip select communicates with the magnetic sensor. The device is essentially the combination of two discrete chips in the same package.

The magnetic measurements are 16-bit signed values and the magnetic field measurement range can be tuned to $\pm 4, \pm 8, \pm 12, \pm 16$. The data rate can also be tuned to multiples of powers of two ranging from $\frac{5}{8}$ Hz to 80 Hz. These properties can be controlled by setting the values in registers on the sensor chips.

The second type of magnetic sensor is the LSM303D [60]. The current prototype has four of these sensors that are used to measure the field generated by the magnetic tracer.

It has the same measurement range as the LSM9DS1, also measured in signed 16-bit values. The data rate is multiples of powers of two ranging from $\frac{25}{8}$ Hz to 100 Hz.

In the proposed application, the sampling rate is expected to sampled at the maximum rate. Because the sampling cannot be exactly aligned between the two chips, there is still a misalignment of up to 2.5 ms. With this in mind, there is an expectation that the change in the observed earth's magnetic field (and other peripheral magnetic field effects) have a limited rate of change.

3.5 Firmware

Firmware was created as part of the SBIR effort. This includes drivers to communicate with the various radio devices as well as the magnetic sensors. The code was developed using a software development kit (SDK) provided by Nordic Semiconductor. I was responsible for the development of the firmware supporting the magnetic sensors as well as the Bluetooth communication. I assisted in the development of the secondary radio firmware. Additionally, the embedded-system implementation of an SVM classifier and a simplified EMF removal algorithm were also implemented by me.

The firmware for the prototype also includes implementation of simple removal of earth's magnetic field as well as a classification algorithm based on a support vector

machine (SVM). The creation of the SVM is implemented by a client program that communicates with the headset. Samples are taken and the raw values are transmitted to a computer for generation of an SVM with a linear kernel. The SVM provides a one-to-one classifier for seven classes and classification is done by a winner-take-all calculation. A matrix and offset are sent back to the headset for the classification to be done on the headset.

On the firmware, the classifier identifies the classes and those are converted to HID commands which are then sent over either a BLE connection or over the secondary wireless communication pathway to be conveyed to a computer, cell phone, or any other suitable device.

For the purposes of this work, it is expected that the raw values are sent to a computer to be processed. The prior work focused around a classification problem. The goal of this research is to facilitate proportional control which requires more sophisticated tracking of the magnetic field values. The target framework, however, remains the same. Configuration and preparation can be done utilizing a computer or a smartphone for configurations. The eventual goal is to allow processing (or tracking of the magnet) on the embedded device on the headset and converting those tracked values to commands to be issued.

The secondary communication pathway was sent to a dongle that was also based on an nRF52 system. The firmware for this device was also designed as part of the SBIR research. This dongle is currently based on a development kit and I wrote the firmware implementing the dongle making use of the radio drivers written by others. The dongle firmware focuses on passing along configuration messages including raw data packets. It also manages the receipt of additional messages intended to convey HID commands. These commands are converted to USB HID commands that can be sent to a smartphone or a computer device.

Since both the BLE and the USB systems create standard HID packets, they can

be used to communicate with supporting devices without the need for specialized drivers. When using the dongle or headset to communicate over a USB connection, it is necessary for the phone or computer to serve as host. This does mean that, when communicating with a smartphone, it is required that the phone support the USB On-the-Go (OTG) standard.

Based on the existing firmware, it is possible to send raw measurement data to be analyzed and processed. The existing system is designed with the intent that configuration settings are performed on external hardware devices and that model information can later be uploaded to the headset for real-time processing of commands. To this end, the supporting algorithms to enable proper tracking will be expected to be performed on a general purpose computer with pieces that are delivered to the headset that can be efficiently computed on embedded systems.

3.6 Client Software

Another element of the system was a client program written in Python that provided a method of communicating with the headset. This system also provided a method for recording measurements from the headset as well as ways to plot the data in a way that could be assessed. Through this effort, experience with interfacing with the hardware and creating communication pathways was established. At this point, the hardware system is to a point where more refined techniques can be employed and there is an established structure provided for doing analysis. The firmware is written in C and C++ with interfacing client code written in Python. To interface with this system, new work will be developed in Python with the intent that future deployment may require code in C or C++.

CHAPTER IV

SYSTEM OVERVIEW

4.1 Tracking Methods

Initial investigation for proportional control focused on methods that allow tracking in nonlinear systems. Nonlinear systems can lead to a variety of challenges. One of these is that they can lead to optimization problems that are not convex. Non-convex problems may lead to finding local minima. In the particular case of tracking a magnet in a magnetic field, there are multiple solutions due to symmetry properties.

Initial work focused on trying to create code to implement particle filtration to try and track the magnet [14]. In investigating this method, a simulated system was created implementing a particle filter in C++ wrapped for implementation from Python with a Cython wrapper. While tracking is possible in simulation, employing such algorithms in real systems is challenging because it requires some pieces of information that is not readily available. One portion is ensuring that the signal measurements are accurate representations of the physical measurements. It is also necessary to have information about the sensor position and the magnetic field strength. This information is necessary to do an assessment for assessing the accuracy of the measurements as compared to the theoretical estimates.

The expression that is used to make an estimate of the expected measurements is given by the magnetic dipole equation.

$$\mathbf{b} = \frac{\mu_0}{4\pi} \frac{3(\mathbf{s}-\mathbf{a})\left((\mathbf{s}-\mathbf{a})\cdot\mathbf{m}\right)}{\|\mathbf{s}-\mathbf{a}\|^5} - \frac{\mathbf{m}}{\|\mathbf{s}-\mathbf{a}\|^3} \tag{1}$$

In this equation, μ_0 represents the permeability of free space. The magnetic moment $\mathbf{m}$ is a vector with an amplitude that is a function of the strength of the magnetic

material and its geometry. It is assumed that the strength of the magnetic moment is available *a priori*. The orientation of the magnet, or the directional component of the magnetic moment, is something to be tracked. The position of the magnet is expressed as **a**. This is also something to be tracked. The important values that must be properly accounted for is the measurement values must calibrated and unbiased, the orientation of the measurement space must match the physical space, and the sensor positions **s** must be known.

An illustration of the elements of the magnetic dipole equation is in Figure 7.

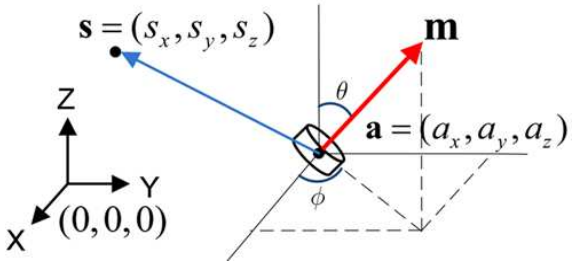


Figure 7. Representation of the magnetic localization problem [23]. The **m** vector represents the magnetic moment vector. **s** represents the sensor location. The **a** vector and angles θ and ϕ represent the position and orientation of the magnet, where the circular faces of the cylindrical magnet are the north and south poles.

Another technique that is expected to be successful is using Kalman filtering. Normal Kalman filters have an underlying linearity assumption which is violated by this problem, but the extended Kalman filter (EKF) provides a method for allowing the Kalamn filter to make an approximation by using a localized linear estimate and a Jacobian to perform updates. Using EKF filters for tracking rotations including using magnetic fields have also been studied demonstrating that they can be used for tracking orientation by using magnetometer data [53]. This work focuses on using earth's magnetic field for the sensing, which is assumed to be a constant planer field. For the headset problem of tracking a local magnet, the field curvature over small regions can be quite large, so EKF methods could perform poorly when the linearization approximation is violated.

An alternative method of Kalman filtering for nonlinear systems uses the Unscented transform. This method, called unscented Kalman filtering (UKF) uses a deterministic rule to select weighted estimates that are spread statistically [70, 71]. This attempts to estimate the second-order statistics to try and give an improved estimate of the measurement. The UKF typically provides more accurate results than the EKF, though it does have some limitations when applied to non-Gaussian distributions. It also has the benefit of not requiring an expression of the Jacobian as is required for the EKF, although the Jacobian is available in this case. It also has the same algorithmic complexity as the EKF. The UKF seems to be a strong candidate for this problem.

Additional methods based on these techniques are also available such as unscented particle filters [67, 52]. These approaches aim to use the benefits of the statistical framework provided by Kalman filtering to improve the estimations of particle filtering while still preserving the generality that particle filters provide to track distributions that are non-Gaussian.

While the investigation of these techniques provided important insights into the available techniques for doing tracking, they all rely on having good models to perform assessment of the goodness of the measurements. This step was the major lacking component. To do this, it is necessary to have some common frame of reference in space and orientation as well as having good a-priori information about sensor positions. Because the headset can be adjusted to suit the user, it is also important that the techniques being employed do not require a lot of specialized equipment or technical background to perform calibration steps. The goal is to create a framework to discover this information through data collection without the need of specialized equipment such as a robotic arm.

4.2 System Configuration

There are several steps that must be accomplished to properly configure the system. The first portion of the architecture requires performing a calibration of the sensors and a relative alignment. The headset provides a magnetic sensor that is far from the tracked magnet for the purpose of removing the Earth's magnetic field (and similar components.) To be able to remove this field component, it is necessary that the sensors be properly calibrated.

Once this magnetic field has been removed, it is necessary to properly orient the sensor measurements to be aligned with the physical space. It is also important to know exactly the positions of the sensors. While some of this information can be approximated by the form of the device when it is constructed, the headset is adjustable.

4.3 Sensor Calibration

It is expected that the magnetic sensors do not necessarily provide truly orthogonal values. To get good measurements, it is important to orthogonalize the measurement values. These measurements also experience some amount of gain and a bias. I order to account for these properties, a set of measurements is recorded in the absence of a magnet to track. It is expected that the measurements sampled by the sensors all represent measurements of the same magnetic field.

In the absence of additional information about the nature of the magnetic fields present at the location of the headset, it is assumed that the magnetic field being observed is dominated by the earth's magnetic field, so the gain for each sensors is tuned according to this value.

To be able to identify the value of this field at a particular location, a model of the Earth's magnetic field is used. A description of this model is provided.

The output of this section is a matrix $\mathbf{A}$ and a vector $\mathbf{o}$ for each of the sensors including the reference sensor. The matrix represents an orthogonalizing and normalizing matrix to ensure that each axial measurement is orthogonal and that each axis has the same gain. The vector serves to remove the bias from the sensor. Once the calibration is performed, raw measurements are processed through an affine transformation. For this portion, each sensor is calibrated independently.

$$\mathbf{b}_{\text{cal}} = \mathbf{A}(\mathbf{b}_{\text{raw}} - \mathbf{o}) \tag{2}$$

4.4 Sensor Alignment

This alignment step focuses on finding a common orientation of calibrated sensor measurements. Again, this processing is taken on data recorded by rotating the headset freely in the absence of a tracking magnet.

To accomplish alignment, the reference sensor is selected to be the common orientation and all the other sensors are aligned to attempt to match data measured by the tracking sensors to match the data from the reference sensor. Once all the sensors are properly aligned, it becomes possible to subtract the component measured by the reference sensor and remove the measurement from the other sensors.

The output from this alignment is a rotation matrix $\mathbf{Q}$ (or alternatively an affine transformation) that attempts to match the measurements for the tracking sensors to the reference sensors. With this alignment, the sensor is rotated to be aligned with the magnetic sensor and the calibrated reference measurements is removed. For data taken in the absence of an external magnet, this value should be approximately zero. In the presence of a localized magnetic field, this should provide an approximation of the magnetic field crated in the presence of the magnet to track.

$$\mathbf{b}_{\text{sense}} = \mathbf{Q}\mathbf{b}_{\text{cal}} - \mathbf{b}_{\text{refcal}} \tag{3}$$

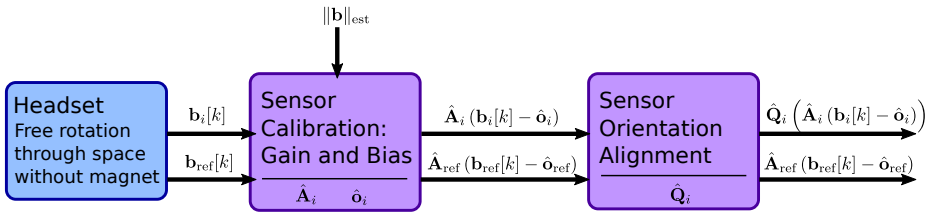


Figure 8. Diagram of the calibration stages. The headset is rotated freely without a magnet present and data are collected. The sensor calibration gain and bias section generates terms of an affine transform, $\hat{\mathbf{A}}_i$ and $\hat{\mathbf{o}}_i$ and outputs calibrated data. Calibrated sensor measurements are passed to a sensor orientation and alignment that generates rotations $\mathbf{Q}_i$ that place all measurements in common orientation with the reference measurements.

The sensor calibration and sensor orientation alignment steps are both performed in a random rotation in the absence of a physical magnet to be tracked. The terms to be generated are $\hat{\mathbf{A}}$ and $\hat{\mathbf{o}}$ for each of the sensors and the reference sensor. The sensor orientation alignment generates a rotation $\mathbf{Q}$ to orient the sensors to align with the calibrated measurements for the reference sensor. Thus the reference sensor has no associated rotation. A diagram of this is included in Figure 8.

4.5 Global Orientation and Sensor Positions

To be able to make forward estimates required for filtering algorithms, it is necessary to take the orientation reference that is provided by the reference sensor and align the measurements into a physical reference frame. At this stage, a magnetic tracer is used. For control, it is expected that this magnet is currently not placed on a person's tongue and that it can be placed statically in a small number of pre-set locations (e.g. five) with a known orientation. The magnet should have a known magnetic moment and the magnet position should be placed in a location that is close to the sensors to provide a strong signal, but far enough away so that the magnetic dipole equation is still a valid approximation (i.e. far-field.)

For this stage, the magnet is placed in a series of pre-defined locations and a set of measurements is taken. Using this information, the model identifies a rotation that

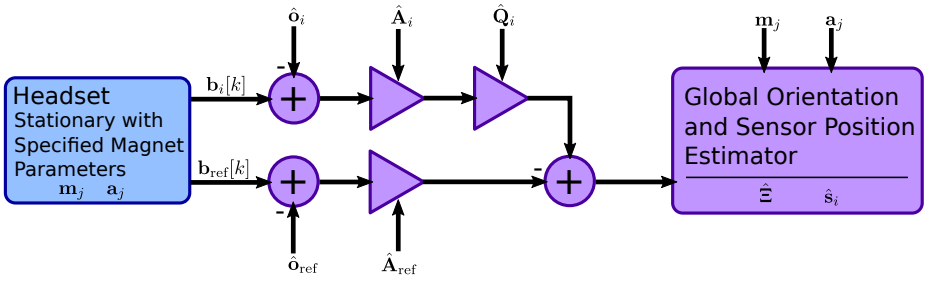


Figure 9. Diagram of the configuration stages. Here the headset is stationary and the magnet is placed in specified positions and orientations with a known magnetic moment strength. Calibration terms are used to remove outside B-field elements. Magnet position and orientation and the B-field measurements are passed to the global orientation and sensor position section that returns a global rotation $\hat{\Xi}$ to rotate sensor measurements to the physical reference frame and sensor position estimates $\hat{\mathbf{s}}_i$.

aligns the tracking sensors to match the global orientation and it identifies the location of the sensors by using the known magnet information. With this, the magnetic field measurements can be processed from the raw measurements and the data can be compared with the known sensor information. A diagram of this portion of the system is in Figure 9.

CHAPTER V

EARTH'S MAGNETIC FIELD

5.1 Magnetic Field Essentials

Papers describing magnetic fields are at times inconsistent with terminology. This section begins by outlining the nature of magnetic field measurements to clarify some of the later points in the content.

Most of the time, the measurements being discussed are based on magnetic flux density, typically notated **B** and frequently called the B-field. It is alternatively referred to as magnetic induction or, with some confusion, it can be called the magnetic field. Magnetic field strength is a closely associated measurement notated **H**, often referred to as the H-field. It is also referred to as the magnetic field intensity, the magnetic field strength, or magnetizing field. It is also often called a magnetic field.

The units of magnetic measurements can also become confusing. The standard unit for **B** is in tesla (T). Tesla is a very large quantity so measurements at the scale presented here are more commonly measured in nT (sometimes denoted in units gamma (γ) which will be avoided here) or in gauss (G) where 1G = 10^5nT. The corresponding magnetic flux $\mathbf{\Phi}_B = \iint \mathbf{B} \cdot d\mathbf{A}$ is the B-field integrated over a surface, denoted **A** and is measured in webers (Wb = Tm^2).

The magnetic field strength **H** is measured in amperes per meter (A/m), or in oersteds (Oe = $\frac{1000}{4\pi}\frac{\mathrm{A}}{\mathrm{m}}$). **B** and **H** are related by the magnetic permeability $\mathbf{B} = \mu\mathbf{H}$. If the measurement is taken in free space, $\mu = \mu_0 = 4\pi 10^{-7}\frac{\mathrm{Tm}}{\mathrm{A}}$ and an H-field of 1 Oe results in a B-field of 1 G, so in some literature, oersteds and gauss are used interchangeably. Since the relative permeability of air is $\frac{\mu_{\mathrm{air}}}{\mu_0} = 1.000\,000\,37$ [17], using the permeability of free space is a good approximation.

Unless otherwise indicated, any discussion of magnetic field is interpreted to mean the B-field and will typically be measured in nT or G.

5.2 Geomagnetic Field Models

The Earth's magnetic field is well-studied. In fact, it has been a topic of research for centuries through measurement and there is a geological record providing even more information to scientists. For the purposes of this effort, it is helpful to know about the Earth's magnetic field as a tool for calibration as well as to allow the removal of the EMF contribution to the magnetic sensors. There are multiple models designed to estimate the EMF values in space and time based magnetic measurements from various locations around the world.

5.2.1 World Magnetic Model

Researchers with the U.S. National Oceanic and atmospheric Administration's National Geophysical Data Center (NOAA/NGDC) and the British Geological Survey (BGS) created a model called the World Magnetic Model (WMM) [15]. The model estimates the Earth's magnetic field over time and is updated every five years. NOAA provides access to their source code [51] as well as an online calculator [6] to provide data based on a particular location (latitude, longitude, and elevation) and at a specific date. The model's limitations are stated to be from 1 km below the Earth's surface to 850 km above the Earth's surface. The geomagnetic field vector B_m is specified by seven elements: the northerly intensity X, the easterly intensity Y, the vertical intensity Z (positive to nadir), the horizontal intensity H, the total intensity F, the inclination angle I and the declination angle D.

The WMM uses spherical harmonics of order 12 for estimating the magnetic potential of the Earth. It is known that the magnetic field changes over time, a phenomenon called geomagnetic secular variation [35] and, to account for this, the WMM uses a linear secular variation adjustment for each spherical harmonic coefficient. Because

the changes are known to be non-linear, the model is updated every five years and a new linear approximation to the non-linear secular variation is applied.

The minimum wavelength resolved by the 12th order spherical harmonics is $\frac{360^\circ}{\sqrt{12\times13}} = 28.8^\circ$ in arc-length. That is 3200 km at the Earth's surface, but this is sufficient to be almost the entire core field and long-wavelength portion of the crustal and oceanic fields.

For the purposes of notation here, the magnetic field is denoted B_m which is a function of the potential field V. In the equations, the geocentric spherical coordinates are the longitude λ, the latitude ϕ', and the radius r. The variable t is time. The relationship between B_m and V is

$$B_m\left(\lambda, \phi', r, t\right) = -\nabla V\left(\lambda, \phi', r, t\right). \tag{4}$$

The potential is expanded in terms of spherical harmonics.

$$V\left(\lambda, \phi', r, t\right) = a\sum_{n=1}^{N}\left(\frac{a}{r}\right)^{n+1}\sum_{m=0}^{n}\left(g_n^m(t)\cos(m\lambda) + h_n^m(t)\sin(m\lambda))\breve{P}_n^m(\sin\phi')\right) \tag{5}$$

Here N represents the degree of the spherical harmonics, $a = 6371200$m is the geomagnetic reference radius, and $g_n^m(t)$ and $h_n^m(t)$ are time-dependent Gauss coefficients of degree n and order m. For any $\mu \in \mathbb{R}$, $\breve{P}_n^m(\mu)$ are the Schmidt semi-normalized Legendre functions.

As this model is based upon the assumption of a linear approximation of the Gauss coefficients, $g(t)$ and $h(t)$, the model is described in terms of coefficients $g(t_0)$ and $h(t_0)$ at an initial reference time $t_0 = 2015.0$ and a value $\frac{dg(t)}{dt}$ and $\frac{dh(t)}{dt}$ in units of $\frac{\text{nT}}{\text{year}}$. With these parameters, the model also specifies uncertainty values and the model is expected to yield values within these uncertainty values over the five-year life of the model.

There are several disturbances that can also contribute to the electromagnetic field at a particular location. The strongest global contribution is from the core field, the field produced by the Earth's liquid-iron outer core. Magnetic materials in the

core can significantly contribute locally to the field. Electric currents flowing through conductive water also create contributions to a magnetic field. This includes materials such as seawater. All these effects are included in the WMM.

Several other disturbance fields also create magnetic fields. Such fields are not represented in the global model, but in an environment with electronic components and with wireless devices generating electromagnetic fields, the ambient magnetic field at a point cannot be completely estimated by the Earth's magnetic field alone.

5.2.2 International Geomagnetic Reference Field

There is another model, the International Geomagnetic Reference Field, which is currently in its twelfth generation [64]. Computationally, it is very similar to the WWM model, but it is defined over a longer period of time. For the epochs up to 1995.0, the coefficients have degree $N = 10$. The IGRF has degree $N = 13$ since that point. Recent models have included data from the low earth orbit (LEO) satellite measurements. The truncation degree was limited to 13 to avoid the inclusion of crustal magnetic field contributions that dominate higher degrees. Detailed practical information including source code in MATLAB is also available [18].

The system is defined using a geocentric system of coordinates, but it can be converted to geodetic coordinates and, in the paper describing the system, the World Geodetic System 1984 (WSG84) [19] is recommended.

5.3 Magnetic Sensors

5.3.1 Sensor Physics

There are several different types of magnetic sensors with different levels of sensitivity and different susceptibilities to noise and other interfering effects. Some of the available options include Hall-effect sensors, magneto-diodes, magneto-transistors, AMR magnetometers, GMR magnetometers, MTJ magnetometers, magneto-optical devices, nuclear precession, optically pumped, fluxgate magnetometers, search coils,

SQUID magnetometers, and MEMS sensors based on the Lorentz force, electron tunneling, and compass-based measurement [40].

An understanding of the magnetic sensors used for taking measurements is beneficial to understanding the nature and challenges of the measurements.

Magnetic sensors are classified as vector magnetometers that measure the vector components of a magnetic field, and total field (or scalar) magnetometers that measure the magnitude of a magnetic field. Of course in this application, the vector components of the sensor measurements are essential.

Many modern, low-cost, compact magnetometers are MEMS-based devices. Most such sensors use the Lorentz force for their measurement, or in other terms, Hall effect sensors.

The Hall effect, discovered by Edwin H. Hall, was discovered by noticing that when a thin rectangular piece of gold foil was placed in a strong magnetic field oriented perpendicular to the surface of the foil, any current sent along the length of the gold was impacted by the Lorentz force. This pushed the flow of electrons toward one side of the foil. This resulted in a difference in potential between the two sides of the foil. This voltage could then be measured to identify the relationship between the field strength and the force exerted on the charged electrons. This force is given by the following equation.

$$\mathbf{F} = q\mathbf{E}_e + q\mathbf{v} \times \mathbf{B} \tag{6}$$

Here $\mathbf{F}$ is a force, q is the charge of a particle, $\mathbf{E}_e$ is the applied electric field, $\mathbf{v}$ is the velocity of the charged particle, and $\mathbf{B}$ is the magnetic field.

From this, the Hall electric field $\mathbf{E}_H$ is approximately given by

$$\mathbf{E}_H \simeq \mu \left[\mathbf{E}_e \times \mathbf{B}\right] \tag{7}$$

where μ is the carrier mobility. This leads to a current deflection effect [49].

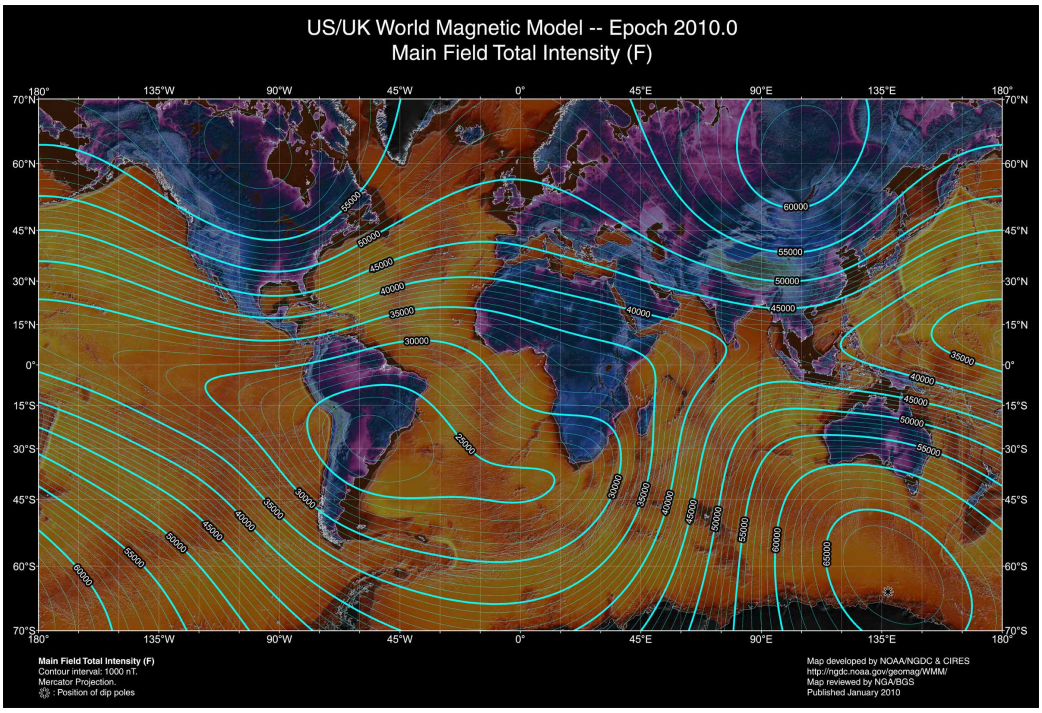


Figure 10. This illustrates contour curves of the strength of the Earth's magnetic field as specified by the WMM model.

The Hall effect is very small in metallic conductors. Within semiconductor material, the effect is much greater. Furthermore, III-V semiconductors have higher electron mobilities than silicon, and therefore have a higher sensitivity. Most such commercial magnetometers use indium antimonide [40].

The Earth's magnetic field on the surface of the earth varies from about 22000 nT to 67000 nT. In gauss (G) that corresponds to between 0.22 G and 0.67 G [15]. Silicon devices have a sensitivity range of 10 to 1000 G, and indium antimonide sensors measure from 10^{-3} G to 1000 G. They can measure constant or time-varying fields with an upper limit of about 1 MHz. Such sensors are light and have low power requirements, making them a good choice for mobile devices.

In MEMS-based magnetometers, the choice of materials for the hard magnet is limited by the need to use hydrofluoric acid (HF) in the release step.

For the purposes of signal processing, the nature and source of the system noise

is of importance for doing proper filtering. Among magnetic sensors, geomagnetic noise, attributed to currents in the ionosphere driven by tidal forces and winds, are of note. This geomagnetic noise exhibits a $1/f$-like frequency spectrum and is on the order of 0.1nT at low frequencies [44]. There are additional $1/f$ noise due to domain wall motion, charge trap sites in the barrier, and near barrier metal interfaces. It is often strongly magnetic-field dependent. MEMS devices can, by means of flux concentrators on MEMS flaps, create an oscillating mechanism with an oscillatory rotation at a specific drive frequency f_m, which ideally should equal the resonant frequencies of the flaps. This creates a modulation signal of the magnetic field at twice the drive frequency ($2f_m$). This allows the measurements to take place at higher frequencies where the $1/f$ noise is much smaller. The signal from the sensor is amplified and the low-frequency signal is extracted through demodulation. This requires the bandwidth to be sufficiently large to carry the measured signal, but the frequency of changing magnetic fields being measured is typically within a few Hz [21].

5.3.2 Sensor Details

The system under test is composed of two different sensors. The first is the ST Microelectronics LSM303D eCompass module [59], and the second is the ST Microelectronics LSM9DS1 sensor [62]. The details of the mechanical components in the devices is not readily available on the datasheets, but based on the size of the package, it can be inferred that they are MEMS-based.

The LSM303D incorporates a tri-axial magnetometer and a tri-axial inertial accelerometer. Additionally, it has a thermometer for temperature measurement, which can be beneficial for adapting the system to temperature fluctuations.

The LSM9DS1 has a tri-axial magnetometer, a tri-axial linear accelerometer, and an angular inertial sensor. It also contains a temperature sensor.

Additional technical details of the magnets are described in Section 3.4.

CHAPTER VI

EMF-BASED SENSOR CALIBRATION

6.1 Aim

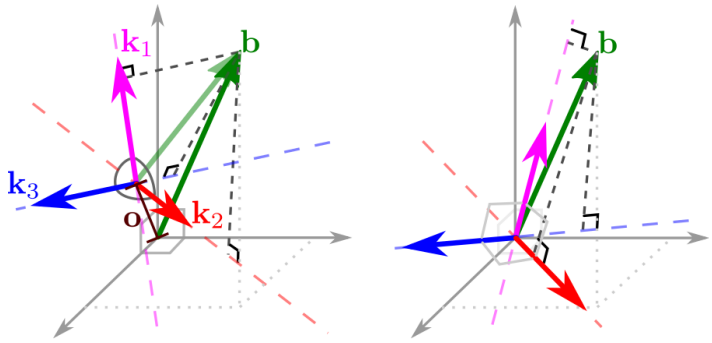
It is necessary to perform a calibration to each of the sensors for accurate performance of the system. Existing techniques were found somewhat lacking for this particular problem, so a novel technique is described based on Quasi-Newton methods.

Proper calibration of sensors is common when using inertial/magnetic sensors and is often used in robotics for managing orientation and position of robotic arms. Calibration is necessary because in a three-axial sensor, it is likely that the three measurement axes are not mutually orthogonal due to manufacturing variances. Furthermore, each of the axial sensors will have a slightly different gain based upon electrical variances in the Hall-effect sensors and device mismatch will lead to some amount of bias on each sensor. Four methods were described in Bonnet et al. [12], namely a Quasi-Newton procedure (QN), Merayo (MER), Minimum-Volume Enclosing Ellipsoid (MVEE), and MinMax (MM) approach where they introduced the MVEE method.

The aim of this system is a little bit different. While relative orientation is important, it is also important to get a sense of the true magnetic field measurements so the magnetic dipole model will generate proper values for evaluation.

6.2 Mathematical Model

The tri-axial magnetic sensors used for this work have three elements that are believed to be orthogonal, but this makes assumptions on the quality of the manufacturing. To get measurements that are correctly calibrated for comparing the measured results



(a) Sensor before calibration (b) Sensor after calibration

Figure 11. A visual depiction of the measurements of a sensor. A field $\mathbf{b}$ is measured by finding the projection of the field onto each of the vectors $\mathbf{k}_i$. These vectors have varying length representing their gains. The offset $\mathbf{o}$ is also depicted. After calibration, the three resulting vectors are orthogonal with a known gain and the offset has been removed.

to an estimated model. Additionally, each sensing element can have a bias and a gain term. A visual depiction of this is illustrated in Figure 11. The measurements can be viewed as finding the projection of the field onto a set of measurement vectors with various lengths. These vectors are also offset by a vector $\mathbf{o}$ representing the bias. To provide a proper measurement, the offset must be subtracted and the vectors must be ajdusted to provide orthogonal vectors with a common gain. This gain is adjusted so that the measurements are representative of physical measurements based on observed magnetic field measurements. It will be important to note that the set of viable solutions to this is only unique up to rotations and reflections. It is assumed that the manufacturer intended that the three axes should be orthogonal and the handedness relationship (e.g. right-handed orientation where $x \times y = z$) is known. These considerations will be revisited when constructing the methods for finding correcting terms to create the orthonormal measurements.

To express this calibration problem mathematically, consider that $\mathbf{b}$ represents the true B-field measurements at a magnetic sensor. Then the measurements taken at that sensor are expressed as

$$\mathbf{b}_{\text{meas}} = \mathbf{K}\mathbf{b} + \mathbf{o}. \tag{8}$$

Here $\mathbf{K} \in \mathbb{R}^{3\times3}$, and $\mathbf{b}$, $\mathbf{b}_{\text{meas}}$, and $\mathbf{o} \in \mathbb{R}^3$. In many applications, sensor calibration is focused on identifying orientations, so the strength of the B-field is not important and the B-field measurements are normalized to have a Euclidean norm of 1.

Conceptually, the measurements $\mathbf{b}_{\text{meas}}$ should all lie on an ellipsoid centered at the point $\mathbf{o}$. The goal of the calibration is to try and identify a matrix and a vector to get back $\mathbf{b}$ from the measurements. Assume that $\mathbf{A} = \mathbf{K}^{-1}$.

$$\mathbf{A}\left(\mathbf{b}_{\text{meas}} - \mathbf{o}\right) = \mathbf{A}\left(\left(\mathbf{K}\mathbf{b} + \mathbf{o}\right) - \mathbf{o}\right) = \mathbf{b} \tag{9}$$

For such a solution to exist, it is necessary that $\mathbf{K}$ be an invertible matrix, i.e. non-singular. For the purpose of this problem, an assumption will be made that the matrix is not even near-singular. This is equivalent to $\det \mathbf{K} \not\approx 0$. Conceptually, $\mathbf{K}$ represents the transformation and scaling of unit vectors in the direction of sensor measurement. This could be expressed in terms of a diagonal gain matrix multiplied by the unit vectors representing the direction of measurement for each directional sensor element $\mathbf{s}_i$.

$$\mathbf{b}_{\text{meas}} = \begin{bmatrix} g_1 & 0 & 0 \\ 0 & g_2 & 0 \\ 0 & 0 & g_3 \end{bmatrix} \begin{bmatrix} \mathbf{s}_1^\intercal \\ \mathbf{s}_2^\intercal \\ \mathbf{s}_3^\intercal \end{bmatrix} \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} + \begin{bmatrix} o_1 \\ o_2 \\ o_3 \end{bmatrix} = \mathbf{G}\mathbf{S}\mathbf{b} + \mathbf{o} \tag{10}$$

Here $\mathbf{K} = \mathbf{G}\mathbf{S}$. For $\mathbf{K}$ to be singular or near-singular means that the three sensors are all coplanar (i.e., $\mathbf{s}_i$ are linearly dependent) or nearly coplanar respectively. Since three-axial sensors are designed intending to measure mutually orthogonal B-fields, it is a safe assumption that $\mathbf{K}$ is invertible.

Assuming that a sequence of measurements is taken at a sensor in the presence of a time-constant magnetic field as the sensor is rotated in space, $\mathbf{b}$ will have constant

amplitude and represent a sphere, so the vectors over time will all lie on the surface of a sphere centered at the origin with radius $\|\mathbf{b}\|_2$. Thus the measurements of $\mathbf{b}_{\text{meas}}$ will all lie on the surface of an ellipsoid centered at $\mathbf{o}$. Furthermore, the principal axes of the ellipsoid will all be at the eigenvectors of $\mathbf{K}$ and the semi-axes are $\{1/\sqrt{\lambda_i}\}_{1\leq i\leq 3}$. This property makes it clear that a singular $\mathbf{K}$ represents a degeneration to an ellipse, a line, or a point.

6.3 Estimating the B-field Radius

It is typical in systems that try to track robotic arms that the magnetic measurements are normalized to unity because the orientation is the important consideration and not the actual field measurements. While it may be possible to use known magnets and controlled environments to identify true magnetic field strengths, such a system is assumed to be unavailable. So rather, this system makes the assumption that the field observed at the sensors is believed to be scaled to $\|\mathbf{b}_{\text{emf}}\|_2$, the strength of earth's magnetic field at the sensor location.

This ground truth is not available directly, but if one assumes that the magnetic field measurements are dominated by Earth's magnetic field, this value can be estimated by using the WMM model. To do this, some geographic information is necessary. The WMM model requires geocentric latitude and longitude and elevation of the location of field measurement.

For the described method, this data was acquired by querying Google geocoding API [29] using an address for the headset under test to get the latitude and longitude and then Google elevation API [28] to get the elevation at that location. Alternatively, if the system is to be implemented on a cell phone or other GPS-enabled device, it may be possible to query the GPS or similar system for the same information. These values are then passed to the WMM model to get the magnetic field information at that point on the Earth.

The assumption that the field measured at the sensor is equal to this value is known to be inaccurate in most circumstances. A magnetic sensor in a typical home or office environment will also pick up distortions from nearby magnetic fields and ferrous materials. It is impossible to account for all these distortions. There are four elements that should be considered here: deviation in field strength from $\|\mathbf{b}_{\text{emf}}\|_2$, change in the direction of the field from $\angle\mathbf{b}_{\text{emf}}$, local deviations in $\mathbf{b}$ at the sensor, and time-varying B-field contributions.

The strength of the field measured at the headset will vary from the WMM model, so the question becomes how does this error impact the system. The result is that the gain of each of the sensors will be too large or small by some factor. While at this stage it is difficult to see how this will impact the whole system, this will cause the estimation of sensor positions (to be found at a later step) to be closer or father away from their true location by some factor and can be accounted for. This will be visited in more detail in 8.

The second consideration is how the angle $\angle\mathbf{b}_{\text{emf}}$ varies from the true observed field at the sensor. This becomes irrelevant because the optimization technique described later is only unique up to some arbitrary rotation, so it has no impact on the system.

Local deviations is a larger problem. This technique assumes that the magnetic field observed at the headset sensors appear to be approximately planar. If the sensor position shifts in space and the magnetic field changes, the sequence of sensor measurements will no longer lie on an ellipsoid. Since this system employs multiple magnetic sensors, such local deviations can result in different sensors measuring different field values. If this effect is significant, the accuracy of the system will be degraded as the reference sensor (located far from the magnet-tracking sensors) is intended to measure and remove the effects of the Earth's magnetic field and other magnetic effects that are not pertinent to the magnet being tracked. Fortunately, this aspect has not seemed to cause significant problems in prior work related to this

headset, so it seems that this type of distortion is small enough that the system is not prevented from operating.

Another factor is if there is a time-varying magnetic field present, it could lead to incorrect calibration of the sensors. In the context of this problem, time-varying magnetic field distortion is treated as noise. The presence of structured noise signal could impact the proper operation of the device, but it would depend on the strength of the field and how it varies over time to see if the proposed system would be able to adequately manage the effects.

6.4 Sensor Calibration

Assuming that the B-field amplitude is assumed to be the Earth's magnetic field as estimated from the WMM model and ignoring additional contributing factors to the measured magnetic field, moving the headset through space in the absence of a tracking magnet should result in points on a sphere. Assume that N measurements are taken as the headset is rotated through space. The earlier model did not account for any noise. To account for this, let ν represent a random variable representing the noise model.

$$\mathbf{b}\text{meas} = \mathbf{GSb} + \mathbf{o} + \nu \tag{11}$$

There is a natural optimization problem to try to find $\hat{\mathbf{A}} \approx \mathbf{A} = (\mathbf{GS})^{-1}$ and $\hat{\mathbf{o}} \approx \mathbf{o}$. This represents an affine transform from the measurements on the ellipsoid to a sphere of radius $\|\mathbf{b}\|_2$. It should be noticed that the affine nature of this problem results in a non-linear, non-convex optimization problem.

The optimization is described by the following minimization, a function described in the literature.

$$\arg\min_{\hat{\mathbf{A}},\hat{\mathbf{o}}} \sum_{i}^{N} \left[\left\| \hat{\mathbf{A}} \left(\mathbf{b}_{\text{meas}} - \hat{\mathbf{o}} \right) \right\|_2 - \|\mathbf{b}_{\text{emf}}\|_2 \right]^2 \tag{12}$$

This function is non-negative, but it is not convex. This objective function attempts to map all the measurements to the surface of a sphere with radius $\|\mathbf{b}_{\text{emf}}\|$. The conversion can converge to a disk-shaped ellipse along the side of a sphere. This can be better understood by extrapolating the formula.

$$\|\mathbf{b}_{\text{emf}}\|_2 \approx \left\|\hat{\mathbf{A}}\left(\mathbf{b}_{\text{meas}} - \hat{\mathbf{o}}\right)\right\|_2 \tag{13}$$

$$\approx \left\|\hat{\mathbf{A}}\left(\mathbf{GSb} + (\mathbf{o} - \hat{\mathbf{o}}) + \nu\right)\right\|_2 \tag{14}$$

In the degenerate case, the matrix $\hat{\mathbf{A}}$ transforms the ellipsoid into a flat disc by $\hat{\mathbf{A}}\mathbf{GSb}$, shifted by $\hat{\mathbf{A}}\left(\mathbf{o} - \hat{\mathbf{o}}\right)$ and the noise $\hat{\mathbf{A}}\nu$ is also squished by the same transformation even further reducing the operand value of the argmin function. Because of this, proper initial conditions is essential to proper convergence.

It can also easily be recognized that this solution for $\hat{\mathbf{A}}$ is not unique, since any rotation and reflection does not change the norm in the optimization when $\hat{\mathbf{o}}$ is an accurate estimate of $\mathbf{o}$.

There are several methods to use to attempt to solve this problem. Several methods were explored by Bonnet et al. including quasi-Newton methods (QN), linearization using the Merayo technique, minimum-volume enclosing ellipsoid (MVEE), and min-max approaches [12]. In their paper, they described different orientations of data collection as well as the sensitivity, offset error, and orthogonality angle error of each technique. They presented a novel MVEE method for performing the optimization, but their method tended to perform worse than the QN and Merayo techniques, which had comparable performance. The details of the QN optimization were not included, but under the conditions ($\mathbf{A}_{ii} > 0, \mathbf{A}_{ij} = 0$ for $i < j$) the resulting matrix is upper-triangular and insures a proper Cholesky decomposition for $\mathbf{A}$.

6.5 Optimization Implementation

A couple of quasi-Newton methods were selected to solve this optimization: the Nelder-Mead (NM) downhill simplex method [43] and the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm [56]. NM only requires a vector of values over which to optimize and an objective function to minimize, but it takes a long time to converge. The BFGS algorithm requires providing the gradient of the objective function which allows more efficient optimization. The implementation used is in the scipy optimize library [20].

The objective function comes directly from the optimization problem. The function takes a vectorized version of $\hat{\mathbf{A}}$ and $\hat{\mathbf{o}}$ as optimization inputs and it takes the measurements $\{\mathbf{b}_i\}_{1\leq i\leq N}$ and $\|\mathbf{b}_{\text{emf}}\|$ as parameters.

$$f(\hat{\mathbf{A}},\hat{\mathbf{o}}) = \sum_{i}^{N}\left[\|\hat{\mathbf{A}}(\mathbf{b}_i-\hat{\mathbf{o}})\|_2 - \|\mathbf{b}_{est}\|\right]^2 \tag{15}$$

The gradient of the objective function is calculated in two parts, one based on $\hat{\mathbf{A}}$ and the other based on $\hat{\mathbf{o}}$. These are vectorized into a 12-dimensional gradient vector to be used for the BFGS.

$$\frac{\partial f(\hat{\mathbf{A}},\hat{\mathbf{o}})}{\partial \hat{\mathbf{A}}} = 2\sum_{i}^{N}\frac{\left\|\hat{\mathbf{A}}\left(\mathbf{b}_i-\hat{\mathbf{o}}\right)\right\|_2 - \|\mathbf{b}_{\text{est}}\|}{\left\|\hat{\mathbf{A}}\left(\mathbf{b}_i-\hat{\mathbf{o}}\right)\right\|_2}\hat{\mathbf{A}}\left(\mathbf{b}_i-\hat{\mathbf{o}}\right)\left(\mathbf{b}_i-\hat{\mathbf{o}}\right)^{\intercal} \tag{16}$$

$$\frac{\partial f(\hat{\mathbf{A}},\hat{\mathbf{o}})}{\partial \hat{\mathbf{A}}} = -2\sum_{i}^{N}\frac{\left\|\hat{\mathbf{A}}\left(\mathbf{b}_i-\hat{\mathbf{o}}\right)\right\|_2 - \|\mathbf{b}_{\text{est}}\|}{\left\|\hat{\mathbf{A}}\left(\mathbf{b}_i-\hat{\mathbf{o}}\right)\right\|_2}\left(\mathbf{b}_i-\hat{\mathbf{o}}\right)^{\intercal}\hat{\mathbf{A}}^{\intercal}\hat{\mathbf{A}} \tag{17}$$

6.5.1 Initial Conditions

To get convergence to a global minimum it is important to start with a good initial estimate. It is also beneficial to have additional information to determine the fitness of the local minima.

It is possible to get a good initial estimate of the final values from the data. If one could assume that the sampled measurements are uniformly distributed around the sphere, it would be possible to estimate the bias by taking the average of the samples. However, as samples are taken as the headset is rotated in space, the samples are found along a curve and are highly unlikely to be nicely distributed.

If the assumption is made that the data points collected for the orientation training are broadly distributed in all directions (though not with any particular distribution), it is still possible to measure the maximum and minimum measurement from each sensor and averaging the result. Taking the maximum and minimum values is highly susceptible to the contributions of noise. More importantly, any anomalous values provided by the sensors can break this algorithm. Poor connections and PCB/placement errors can make this problem even worse.

One of the easiest ways to overcome this problem is to use a median filter on the data. Hereafter, a median filter operating on a signal X with a length L will be denoted $\text{med}(X, L)$. Because the values are vector-based, the median operation is built on the norm of the vectors.

Once a median filter has been used on the data accounts for removing any isolated errors. Additionally, the system can take an average, a median, or select a certain number of samples to ignore to further eliminate outliers.

This technique does require that the earth's magnetic field is placed nearly collinear with each of the axes of measurement. If the headset is only rotated along a hemisphere rather than a complete sphere, this method would lead to the offset being placed central to the sampled hemisphere rather than at the center of the ellipsoid as intended. This requires that the user rotate the headset through many orientations to get good coverage of the sphere. These measured values are then used as the initial estimate for the bias $\hat{\mathbf{o}}_0$.

$$\hat{\mathbf{o}}_0 = \left\{ \frac{\max_i([\mathbf{b}_i]_j) + \min_i([\mathbf{b}_i]_j)}{2} \right\}_{1 \leq j \leq 3} \tag{18}$$

To estimate the matrix $\hat{\mathbf{A}}$, it could be assumed that the sensors are mutually orthogonal resulting in $\hat{\mathbf{A}}$ being the product of a rotation matrix and a diagonal matrix. Again an assumption is made that the Earth's magnetic field is rotated enough to be nearly collinear with each measurement axis. By subtracting the earlier calculated bias, the radius can be estimated again using the max values.

$$\hat{\mathbf{A}}_{j,j} = \frac{\|\mathbf{b}\|}{\max_i \left([\mathbf{b}_i]_j\right) - [\hat{\mathbf{o}}]_j} \tag{19}$$

$$\hat{\mathbf{A}}_{i,j} = 0 \quad \text{for } i \neq j \tag{20}$$

6.5.2 Upper-Triangular Factorization

Because the objective function is spherically symmetric, the solution of $\hat{\mathbf{A}}$ is unique up to an arbitrary rotation. To provide some consistency to the estimation, it is possible to do a QR factorization of the system to find a suitable upper-triangular matrix.

A QR factorization factors the matrix into the product of a unitary matrix $\mathbf{Q}$ and an upper triangular matrix $\mathbf{R}$ [42]. In this case, $\hat{\mathbf{A}}$ is a real-valued square matrix. Furthermore, it is assumed that the relative orientation of the three sensor axes are approximately orthogonal and form a right-hand oriented space, i.e. $\det \mathbf{A} = +1$. This indicates that, given good initial conditions, $\mathbf{Q}$ will be a rotation. It is a certainty that $\mathbf{Q}\mathbf{Q}^H = \mathbf{I}$ where $\mathbf{I}$ is an identity matrix and ^{H} denotes the Hermitian conjugate, meaning that $\mathbf{Q}$ is unitary. The matrices under consideration are real-valued, so this is equivalent to a transpose meaning that $\mathbf{Q}$ is an orthogonal matrix.

Using an Upper-Triangular Factorization can save a little bit of storage and can be used to speed up calculations. Some of the properties of upper-triangular matrices

also can be beneficial, such as the fact that the eigenvalues of a triangular matrix are found on the diagonal. However, in the proposed process, it is possible to combine most of the matrix calculations in the processing and with the added rotation, this factorization is not necessary.

6.5.3 Recognizing Degenerate Optimizations

It is useful to recognize when the optimization fails to represent the convergence of the system to the proper spherical shape but rather into a degenerate disk on the surface of the zero-centered sphere instead.

To recognize this, the determinant of the matrix $\hat{\mathbf{A}}$ as the degenerate case will lead to the determinant approaching 0. An estimate of the accurate determinant can be understood by recognizing that the bounding rectangle containing the ellipsoid of measurements has the following upper limits u and lower limits l.

$$u_j = \max_i [\mathbf{b}_i]_j \tag{21}$$

$$l_j = \min_i [\mathbf{b}_i]_j \tag{22}$$

The volume of this bounding box, expressed as $V_{\mathbf{b}_i}$ is

$$V_{\mathbf{b}_i} = \prod_j (u_j - l_j)\,. \tag{23}$$

The matrix $\hat{\mathbf{A}}$ is designed to optimized to map the ellipsoid to a sphere of radius $\|\mathbf{b}_{\text{emf}}\|$ contained in a bounding box with volume approximately equal to

$$V_{\mathbf{b}_{\text{emf}}} = 2^3 \|\mathbf{b}_{\text{emf}}\|_2^3. \tag{24}$$

Because $V_{\mathbf{b}_i}$ is calculated by using the strict limits, it is highly likely that $\det \hat{\mathbf{A}} \leq V_{\mathbf{b}_{\text{emf}}}/V_{\mathbf{b}_i}$. The degenerate case where the sphere is mapped to a flattened disk will result in $\det \hat{\mathbf{A}} \ll V_{\mathbf{b}_{\text{emf}}}/V_{\mathbf{b}_i}$. There is also another possibility that is most likely when

the data sampled does not adequately cover the surface of the sphere where a larger ellipsoid could be mistaken for the correct adjustment. In such a case, there is a possibility that the determinant may be larger.

Then let $0 < \kappa < 1$ represent a threshold parameter ($\kappa > 0.9$ is a good value.) If the following inequality holds true, $\hat{\mathbf{A}}$ is accepted. If not, either more data can be collected or initial conditions of the optimization can be used. A negative determinant indicates a reflection and can easily be fixed.

$$\det \hat{\mathbf{A}} > \kappa \frac{V_{\mathbf{b}_{\text{emf}}}}{V_{\mathbf{b}_i}} \tag{25}$$

6.5.4 Processing the Data

At this point, the data is processed as $\mathbf{b}_c$, the calibrated data is calculated using the following equation.

$$\mathbf{b}_c = \hat{\mathbf{A}}(\mathbf{b}_i - \hat{\mathbf{o}}) \tag{26}$$

Values processed by this method should have terms which are mutually orthogonal measurements of the B-field under some arbitrary rotation scaled by the estimate of the B-field measured at the sensor, assumed to be approximately equal to the Earth's magnetic field.

6.6 Ellipse-Fitting Method

The prior method was based around trying to find the inversion of the matrix $\mathbf{A}$ as outlined by the model and an estimate of the offset $\mathbf{o}$. A method of approximating the initial conditions was presented. In this alternative approach, a technique that appears to be a novel method, the system attempts to fit the data to an ellipsoid and then a transformation matrix is found to transform that data to the appropriate spherical shape. While this method is again based on the same optimization

algorithms as the prior section, it is more founded in linear algebra.

6.6.1 Quadric Equation of an Ellipsoid

An arbitrary ellipsoid centered at $\mathbf{o}$ can be defined by a quadric relationship.

$$\{\mathbf{x}|(\mathbf{x}-\mathbf{o})^{\intercal}\mathbf{E}(\mathbf{x}-\mathbf{o})=1\} \tag{27}$$

This holds for some $\mathbf{E}$ that is positive-definite. It has the useful property that the eigenvalues of $\mathbf{E}$ represent the reciprocals of the squares of the semi-axes of the ellipse, and the corresponding eigenvectors are unit vectors in the direction of the corresponding principal axes of the ellipsoid.

Using the mathematical framework for the gain and offset of the system, it is possible to express the values of $\mathbf{x}$ in those terms.

$$\mathbf{x}=\mathbf{K}\mathbf{b}+\mathbf{o}+\nu \tag{28}$$

For the optimization, this can be used to identify an objective function over which to optimize.

6.6.2 Quadric Optimization

For simplicity, assume that the true magnetic field measurements $\mathbf{b}$ are assumed to be taken on a unit sphere, i.e. the $\|\mathbf{b}\|_{\text{emf}}$ is assumed to be factored out. Plugging in the expression for $\mathbf{x}$ into the quadric equation results in

$$[\mathbf{K}\mathbf{b}+\mathbf{o}+\nu-\hat{\mathbf{o}}]^{\intercal}\,\mathbf{E}\,[\mathbf{K}\mathbf{b}+\mathbf{o}+\nu-\hat{\mathbf{o}}]=1. \tag{29}$$

Ignoring the effects of noise ν and assuming that the estimate of the center of the ellipsoid $\hat{\mathbf{o}}\approx\mathbf{o}$, this equation reduces to

$$\mathbf{b}^{\intercal}\mathbf{K}^{\intercal}\mathbf{E}\mathbf{K}\mathbf{b}=1. \tag{30}$$

Because $\mathbf{b}$ is assumed to be a unit vector in $\mathbb{R}^3$, this relationship is true when $\mathbf{K}^\intercal\mathbf{E}\mathbf{K} = \mathbf{I}$. Solving this function for $\mathbf{E}$ gives the following result.

$$\mathbf{E} = (\mathbf{K}\mathbf{K}^\intercal)^{-1} = \mathbf{K}^{-\intercal}\mathbf{K}^{-1} \tag{31}$$

This is a useful result as it indicates that $\mathbf{E}$ is a symmetric matrix. Let $\hat{\mathbf{E}} = \mathbf{K}^{-1}$ making the approximation of $\mathbf{E} = \hat{\mathbf{E}}^\intercal\hat{\mathbf{E}}$.

There are two different optimizations that can be used. The first one attempts to find $\mathbf{E}$ directly, and the other attempts to find $\hat{\mathbf{E}}$. Both methods can be employed successfully, but the second one builds on the idea that $\mathbf{E}$ is a symmetric matrix and so can be factored. Finding $\mathbf{E}$ explicitly proved to be more unstable.

6.6.2.1 Optimizing Over Quadric Matrix

The objective function over $\mathbf{E}$ is given by the following equation. It is necessary to square the estimate to ensure that the individual terms of the sum are non-negative.

$$F_{\text{obj},1}(\mathbf{E}, \hat{\mathbf{o}}) = \sum_k \left[(\mathbf{x}_k - \hat{\mathbf{o}})^\intercal\mathbf{E}(\mathbf{x}_k - \hat{\mathbf{o}}) - 1\right]^2 \tag{32}$$

Again, there are two Jacobian functions, one with respect to $\mathbf{E}$ and the other with respect to $\hat{\mathbf{o}}$.

$$\frac{\partial F_{\text{obj},1}}{\partial \mathbf{E}} = 2\sum_k \left[(\mathbf{x}_k - \hat{\mathbf{o}})^\intercal\mathbf{E}(\mathbf{x}_k - \hat{\mathbf{o}}) - 1\right](\mathbf{x}_k - \hat{\mathbf{o}})(\mathbf{x}_k - \hat{\mathbf{o}})^\intercal \tag{33}$$

$$\frac{\partial F_{\text{obj},1}}{\partial \hat{\mathbf{o}}} = -2\sum_k \left[(\mathbf{x}_k - \hat{\mathbf{o}})^\intercal\mathbf{E}(\mathbf{x}_k - \hat{\mathbf{o}}) - 1\right]\left[(\mathbf{x}_k - \hat{\mathbf{o}})^\intercal\mathbf{E} + (\mathbf{x}_k - \hat{\mathbf{o}})^\intercal\mathbf{E}^\intercal\right] \tag{34}$$

6.6.3 Optimizing Over Factorized Quadric Matrix

This method has a couple of benefits over the previous method. The first benefit is that it enforces that the matrix $\mathbf{E}$ will be symmetric by being the product of a real-valued matrix with its transpose.

$$F_{\text{obj},2}(\hat{\mathbf{E}}, \hat{\mathbf{o}}) = \sum_k \left[(\mathbf{x}_k - \hat{\mathbf{o}})^\intercal \hat{\mathbf{E}}^\intercal \hat{\mathbf{E}} (\mathbf{x}_k - \hat{\mathbf{o}}) - 1 \right]^2 \tag{35}$$

$$= \sum_k \left[\left\| \hat{\mathbf{E}} (\mathbf{x}_k - \hat{\mathbf{o}}) \right\|^2 - 1 \right]^2 \tag{36}$$

For brevity in the notation, the following substitution is used.

$$\Gamma_k = \left\| \hat{\mathbf{E}} (\mathbf{x}_k - \hat{\mathbf{o}}) \right\|^2 - 1 \tag{37}$$

This gives the following Jacobian calculations.

$$\frac{\partial F_{\text{obj},2}}{\partial \mathbf{E}} = 4 \sum_k \Gamma_k \hat{\mathbf{E}} (\mathbf{x}_k - \hat{\mathbf{o}})(\mathbf{x}_k - \hat{\mathbf{o}})^\intercal \tag{38}$$

$$\frac{\partial F_{\text{obj},2}}{\partial \hat{\mathbf{o}}} = -4 \sum_k \Gamma_k (\mathbf{x}_k - \hat{\mathbf{o}})^\intercal \hat{\mathbf{E}}^\intercal \hat{\mathbf{E}} \tag{39}$$

6.6.4 Finding Initial Conditions

Just like the prior technique, the objective function is non-convex and so initial conditions are important to getting a correct optimization.

The same methods that were used before can be used here for the estimation of the offset $\hat{\mathbf{o}}$, but it is possible using some statistics from the data to estimate the initial matrix $\mathbf{E}$.

Let B be a random vector of uniformly distributed values on the surface of an n-dimensional sphere. The variance of this random vector is

$$\text{Var}B = \frac{1}{n} \mathbf{I}_n. \tag{40}$$

This property comes from the same essential technique used for generating values uniformly distributed over the surface of an n-dimensional hypersphere. Given a

random vector Y with elements $Y_i \sim \mathcal{N}(0, 1.)$ which are mutually independent random variables, the random vector B is defined as

$$B = \left[B_i = \frac{[Y_i]}{\|Y\|} \right]_{1 \leq i \leq n}. \tag{41}$$

Because of symmetry of a unit sphere, $\mathbb{E}(B_i B_j) = -\mathbb{E}B_i B_j$ for $i \neq j$. This means the covariance terms are all zero. The diagonal entries can be calculated by the variances of the elements of the vector.

$$\sum_i \mathbb{E}\left(B_i^2\right) = \mathbb{E}\left(\sum_i B_i^2\right) = 1. \tag{42}$$

Equality to one is due to the nature of the fact that the vectors lie on the surface of a unit sphere. Symmetry again is used to verify that each of the terms is identical, so each must be equal to $\frac{1}{n}$. For the purposes of this, let Σ_B represent this variance.

The covariance of the transformed vector $\mathbf{x} = \mathbf{K}\mathbf{b} + \hat{\mathbf{o}}$ can be calculated as well. Let X represent the random vector from which $\mathbf{x}$ is sampled.

$$\Sigma_X = \mathrm{Var}(\mathbf{X}) = \mathrm{Var}\left(\mathbf{K}B + \hat{\mathbf{o}} + \nu\right) = \mathbf{K}\Sigma_B\mathbf{K}^\intercal + \mathrm{Var}(\nu) = \frac{1}{n}\mathbf{K}\mathbf{K}^\intercal + \Sigma_\nu \tag{43}$$

This value can be measured from sampling the data. The variance of the noise term ν is dependent upon the nature of the noise. If it is additive and uncorrelated between the different sensor axes measurements, it would be a diagonal matrix with the strengths of the noise. A more accurate estimate of the model of the noise can be estimated by sampling.

The optimization aims to find $\hat{\mathbf{E}} = \mathbf{K}^{-1}$ or alternatively $\mathbf{E} = \mathbf{K}^{-\intercal}\mathbf{K}^{-1}$, it is possible to generate an estimate from the covariance of the measurements. After subtracting the value of the variance matrix of the noise ν, multiply the covariance by n, the result is the inverse of $\mathbf{E}$.

$$\mathbf{E} \approx [\mathbf{n}\boldsymbol{\Sigma}_{\mathbf{k}} - \boldsymbol{\Sigma}_{\nu}]^{-1} \tag{44}$$

Because of the nature of the matrix $\mathbf{E}$, it can be interpreted as having eigenvectors that are the inverse-square values of the semi-axes of the ellipse with unit eigenvectors in the direction of the principal axes of the ellipsoid. By its nature, the variance matrix is positive semi-definite (and in this case it is very likely to be positive definite) and symmetric. This means that $\hat{\mathbf{E}}$ exist such that $\mathbf{E} = \hat{\mathbf{E}}^{\intercal}\hat{\mathbf{E}}$. One natural way to find this matrix is to do an eigenvalue decomposition of $\mathbf{E}$.

$$\mathbf{E} = \mathbf{Q}\Lambda\mathbf{Q}^{-1} = \mathbf{Q}\Lambda\mathbf{Q}^{\intercal} \tag{45}$$

$$\hat{\mathbf{E}} = \sqrt{\Lambda}\mathbf{Q}^{\intercal} \tag{46}$$

Here, $\sqrt{\Lambda}$ represents the diagonal matrix with entries $\sqrt{\lambda_i}$ for the corresponding eigenvalues. Note that the reason that it is possible to do relace $\mathbf{Q}^{-1}$ with $\mathbf{Q}^{\intercal}$ is because the eigenvectors of a real symmetric matrix are orthogonal.

Another way to decompose $\mathbf{E}$ is using a Cholesky decomposition. The Cholesky decomposition finds an upper-triangular matrix $\mathbf{U}$ such that

$$\mathbf{E} = \mathbf{U}\mathbf{U}^{H} \tag{47}$$

Because $\mathbf{E}$ is a real, symmetric matrix, the matrix $\mathbf{U}$ exists and is a real-valued upper-triangular matrix. When the matrix is positive definite, this decomposition is unique [27].

One of the benefits of the Cholesky decomposition is that it is numerically stable, even with no pivoting [26].

6.6.5 Processing the Data

The processing here is done in the same manner as with the previous algorithm. The optimization will either return $\mathbf{E}$ or $\hat{\mathbf{E}}$ and an offset estimate $\hat{\mathbf{o}}$.

Because the analysis made the assumption that the values were to be normalized to unit vectors, an additional multiplication by the estimated value of the Earth's magnetic field at the point is required.

If the optimization provided a value of $\mathbf{E}$, it can be decomposed either by the eigenvalue decomposition method or the Cholesky decomposition as described in the previous section. Let $\mathbf{U}$ represent this resulting matrix factor.

$$\mathbf{b}_c = \|\mathbf{b}_{\text{emf}}\|\mathbf{U}(\mathbf{b}_i - \hat{\mathbf{o}}) \tag{48}$$

If the second method is used and $\hat{\mathbf{E}}$ is found, it can be used directly.

$$\mathbf{b}_c = \|\mathbf{b}_{\text{emf}}\|\hat{\mathbf{E}}(\mathbf{b}_i - \hat{\mathbf{o}}) \tag{49}$$

6.7 Covariance-Based Fitness Tests

The variance analysis used in the second method provides an improvement in fitness over the previous estimation by finding the bounding box of the data. The smallest bounding box of an ellipsoid has sides twice the length of the semi-axes. This means that the determinate of the matrix $\mathbf{A}$, or in the later algorithm $\hat{\mathbf{E}}$ can be estimated using the covariance matrix of the measurements.

$$\det(\mathbf{A}) \approx \frac{\|\mathbf{b}_{\text{emf}}\|^3}{\sqrt{\det(3\Sigma_X - \Sigma_\nu)}} \tag{50}$$

$$\det(\mathbf{E}) \approx \frac{\|\mathbf{b}_{\text{emf}}\|^6}{\det(3\Sigma_X - \Sigma_\nu)} \tag{51}$$

$$\det(\hat{\mathbf{E}}) \approx \frac{\|\mathbf{b}_{\text{emf}}\|^3}{\sqrt{\det(3\Sigma_X - \Sigma_\nu)}} \tag{52}$$

To provide limits to the test, use parameters $u, l \in (0, 1)$ representing the lower and upper tolerances of the determinant. Both values should be close to 0.

$$1 - l < \frac{\det(\mathbf{A})\sqrt{\det(3\Sigma_X - \Sigma_\nu)}}{\|\mathbf{b}_{\text{emf}}\|^3} < 1 + u \tag{53}$$

$$1 - l < \frac{\det(\mathbf{E})\det(3\Sigma_X - \Sigma_\nu)}{\|\mathbf{b}_{\text{emf}}\|^6} < 1 + u \tag{54}$$

$$1 - l < \frac{\det(\hat{\mathbf{E}})\sqrt{\det(3\Sigma_X - \Sigma_\nu)}}{\|\mathbf{b}_{\text{emf}}\|^3} < 1 + u \tag{55}$$

In these equations, known covariance matrices could be provided, or they could be collected by sampling.

6.8 Evaluation

6.8.1 Metrics

For evaluating this metric, there are a few pieces to consider.

- Offset accuracy
- Roundness
- Radius

The first is how accurately the model matches the bias values with $\hat{\mathbf{o}}$. The metric for this is the Euclidean distance $\mathbf{o} - \hat{\mathbf{o}}$.

The second is how spherical the set of measurements happens to be. This is equivalent to how close $\hat{\mathbf{A}}\mathbf{K}$ is to a member of SO(3), the set of rotation matrices. To find this value, the difference between the largest and smallest eigenvalue amplitudes (i.e. complex modulus) of the matrix $\hat{\mathbf{A}}\mathbf{K}$ is found. If the shape is perfectly round, this value will be zero. Alternatively, the ratio of the values can be taken and compared to unity.

The third metric is the radius. It should be the same as the underlying magnetic field strength measured at that point, but due to noise it may be slightly larger

or smaller. This value is calculated by taking the absolute difference between the spectral radius and the estimated B-field strength, i.e. $||\lambda_1| - \mathbf{b}_{\text{emf}}|$.

These three values can be combined in a weighted sum to give a single metric of goodness.

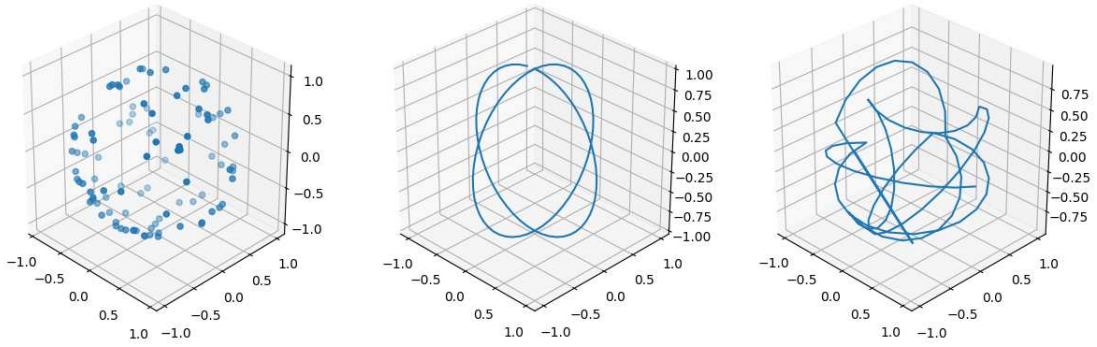
6.8.2 Methods

For this experiment, random rotation matrices is created by generating a vector of eigenvalues $\boldsymbol{\lambda}$ with independent, identically-distributed (i.i.d.) log-normal distributions with $\mu = 0$ and $\sigma = \frac{1}{2}$. Then three random eigenvectors are selected from a uniform distribution on unit vectors. Then the matrix is calculated by $\mathbf{K} = \mathbf{Q}\text{diag}(\boldsymbol{\lambda})\mathbf{Q}^{-1}$. This results in a positive definite matrix almost surely (because there is a chance that the eigenvectors may not be linearly independent.) In practice, there is a small chance that the resulting matrix may be singular or nearly singular, but this chance is accepted and no a-priori effort is made to test for this condition. It is just expected that some optimizations may fail.

The offsets of the vectors are distributed with components according to a Gaussian distribution with $\mu = 0$ and $\sigma = \frac{1}{2}\|\mathbf{b}_{\text{emf}}\|$. Because the results of the simulation are essentially equivalent up to a scaling, the assumption is made that $\|\mathbf{b}_{\text{emf}}\| = 1$.

Because the amplitude of the estimated B-field amplitude only acts as a scaling parameter, it is assumed in the simulation testing that the B-field strength is $\|\mathbf{b}_{\text{emf}}\| = 1$.

For the generation of data points, three different types of data points were generated. The first is using a random cloud of points that exist on the surface of the unit sphere. The second type does a full rotation about the y axis followed by a full rotation about the x axis. The third type of data, which is the most realistic, is a random walk to simulate a person moving the headset freely. To generate this random walk, a velocity of rotation was chosen at $2\pi/30$ radians per sample. Then a random rotation



(a) Uniformly-distributed random points

(b) Deterministic axis rotation path

(c) Random walk path

Figure 12. Illustration of simulated magnetic field measurements previously denoted as $\mathbf{b}$ before the application of the distortion $\mathbf{K}$ and the offset $\mathbf{o}$. These values represent a ground truth field strength with the assumption that the strength of the magnetic field is unity.

vector was selected by finding a random unit vector in $\mathbb{R}^3$. The rotation was then applied for a number of steps selected by $n_{\text{step}} = 15\,\text{Lognormal}(0, 0.25)$. This has a mean just over 15 with a median of 15. Illustrations of the types of data generated are depicted in Figure 12.

For generating testing values, a long set of 100,000 points was generated for each of the path types. For the uniformly distributed random points and the random walk path, a contiguous subset of N random points was selected from within the larger set. For the deterministic path, the values were evenly distributed along the curves outlined representing a controlled testing path environment as might be realized by a robotic arm.

The values generated using these methods were then distorted by the matrix $\mathbf{K}$ and offset by $\mathbf{o}$ and those vectors were provided for the testing.

Simulations were run using all three of the methods described in this chapter. For each experiment, the results of the experiments were saved in a database to be queried. The number of points used were specified by $\{10, 30, 100, 300, 1\,000, 3\,000, 10\,000\}$. Noise was added with a standard deviation of $\{0, 0.00316, 0.01, 0.0316, 0.1, 0.316, 1\}$. This corresponds to signal-to-noise ratios of $\{50\text{dB}, 40\text{dB}, 30\text{dB}, 20\text{dB}, 10\text{dB}, 0\text{dB}\}$.

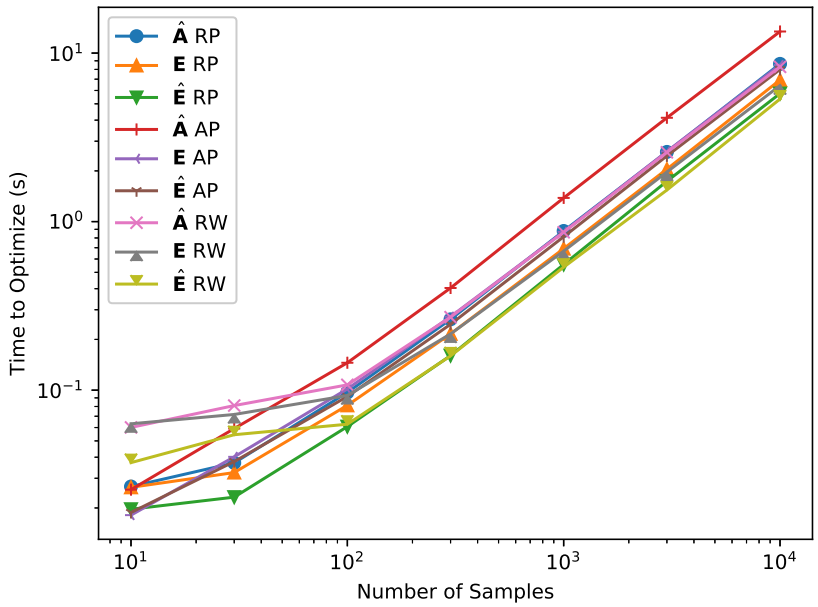


Figure 13. Depiction of the average runtime using each of the three methods and given each of the three path types. In the legend, RP indicates random points, AP indicates axial paths, and RW indicates random walks.

6.8.3 Execution Time

The amount of time to run the optimization is shown in Figure 13. The time to run the optimization is roughly linear. The number of data points was selected logarithmically. The amount of time to optimize is uncertain for low numbers of samples. This is expected as the problem can become poorly defined, especially for small numbers of data points and in the presence of noise. This is especially true for the random walk method, since the continuous random walk is highly likely to be a poor representation for fewer than 100 points, which can be seen by the increased runtime.

Using 10 or 30 points yields poor results and often fails to achieve the convergence constraint within a reasonable number of iterations. For 100 points, the time to optimize is approximately one-tenth of a second. For 10,000 samples, the runtime is still on the order of seconds. Because this sensor calibration is expected to be run

rarely, this is easily acceptable.

Using the method estimating the matrix $\hat{\mathbf{A}}$ is the least efficient method by a factor of about 2 for the axial path method and about 3/2 for random-point and random-walk methods. The $\hat{\mathbf{E}}$ is the most efficient method, though it is quite close to the $\mathbf{E}$ method in efficiency.

Another interesting result is that the axial path method gives less efficient calculation than when the data follows a random sampling or random walk path. As will be seen later, this seems to be because this path is less well-defined and consequently the optimization algorithm fails to converge as quickly. Another indicator of this is how, for a low number of points, a random sampling is most efficient.

All three methods are dependent upon the accuracy required by the simulation. In this case, the stopping condition is that the norm of the Jacobian is less than 10^-8 for all methods.

6.8.4 Metrics Results

To test the success of the system, the results of the optimizations are tested using the proposed metrics.

An increasing number of data points leads to increased accuracy in the estimates of the given metrics, as is expected. With increased noise, the error increases, as would also be expected. In the plots below, points are omitted when the optimization failed to converge. This result is reported by the BFGS algorithm. The results make no attempt to discard values that fail the fitness tests described in this chapter.

If the tests were unsuccessful in generating estimates, the points are omitted. This is typical for small amounts of data or when there is a large amount of noise and convergence does not occur with enough data.

The noiseless case is not included on the plots as errors on the noiseless case can get quite a bit lower (on the order of 10^{-8}) with as few as 100 samples. This provides

a good rule-of-thumb for the fewest number of points that should be used for these techniques. This is true across all metric terms and is due to the stopping criterion of the optimizations and the precision of the computations. While these results show that the methods work as intended, the particular metrics for the noiseless case are not otherwise particularly meaningful.

The case where the noise is as strong as the signal is also not included as the results are generally not good enough to be used practically and many samples are required for proper convergence. This is, however, a somewhat informative result. Intuitively, this can be taken to mean that when the noise is strong enough, the data no longer appears to be an ellipsoid and the described methods begin to break down. With enough data points, these systems can converge, but they do so poorly.

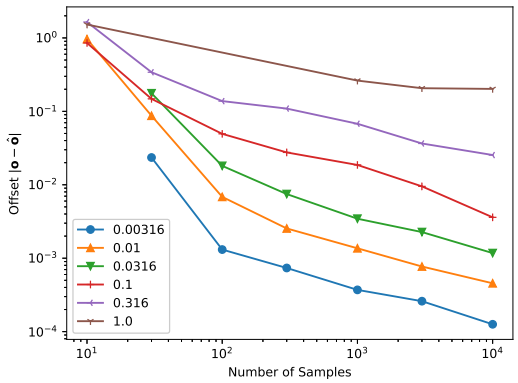
While three types of data was generated, the results in the next section only describe the values from the random walk method unless otherwise specified. This is because the random walk is the most similar to the expected use case scenario.

6.8.4.1 Bias Estimation

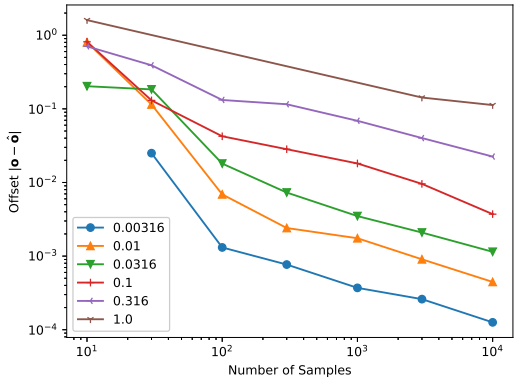
The bias values as estimated by the three described techniques are depicted in Figure 14. The offset estimates seem to continue to improve with increasing data. It seems that 100 samples is approximately the lower limit of acceptable number of data points to give good convergence. Offset estimation is comparable across all three methods. As the amount of noise increases, the number of samples required increases to about 3,000. At a sample rate of 100 Hz, this is 30 seconds of data, which is fairly reasonable.

6.8.4.2 Roundness Estimation

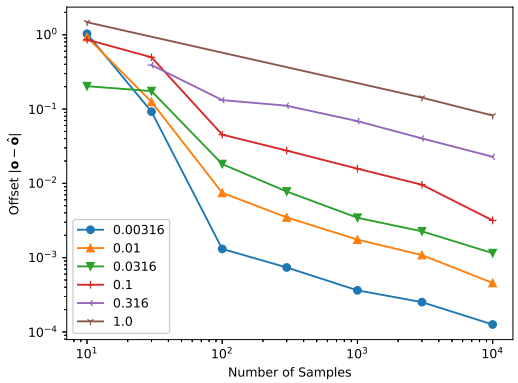
The roundness is a comparison between the modulo of the eigenvectors of the product of the adjustment matrix ($\hat{\mathbf{A}}, \sqrt{\mathbf{E}}, \hat{\mathbf{E}}$), and the distortion matrix ($\mathbf{K}$.) Ideally, this matrix should be a member of the special orthogonal group SO(3), i.e. a rotation, and



(a) $\hat{\mathbf{A}}$



(b) $\mathbf{E}$



(c) $\hat{\mathbf{E}}$

Figure 14. Estimates of $\|\mathbf{o} - \hat{\mathbf{o}}\|$ using each of the three different optimization techniques using random walk samples. Labels are indicative of the method used for optimization, but the corresponding matrix is not directly related to the offset estimate.

all the eigenvalues should have an amplitude equal to one (they could be complex-valued.) In such a case, the roundness value is zero.

A plot of the roundness is found in Figure 15 in the left column.

Roundness seems to be one value that can increase with increasing sample counts. When there is a lot of noise with few datapoints, it is more likely for an optimization to fail. This leads to fewer samples being used in the plots making the plotted values less statistically significant. Some of the roundness, however, seems that there may be more leading to a lack of monotonicity. This seems to occur only for large quantities of noise. So far an explanation has not been identified.

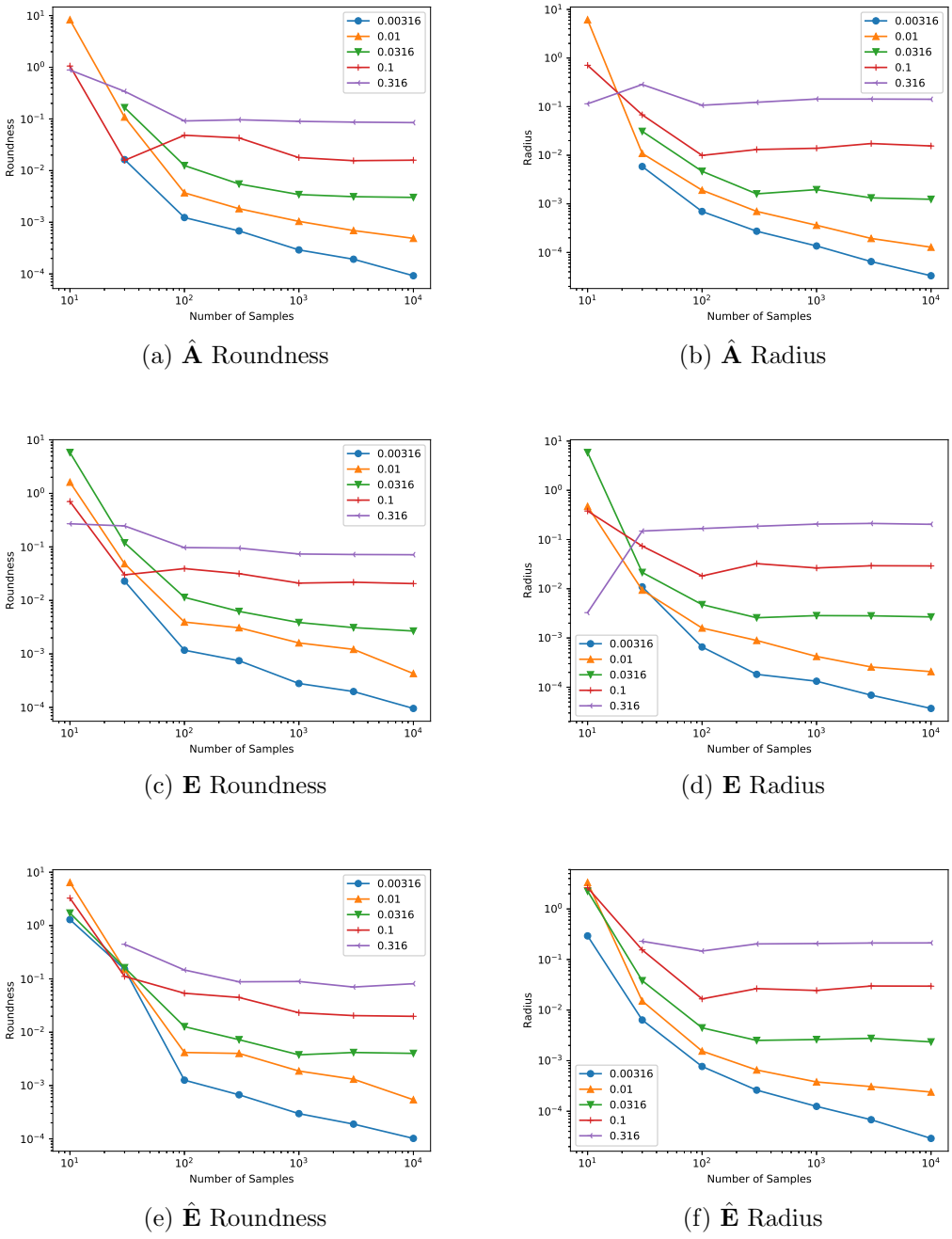
6.8.4.3 Radius Estimation

The radius estimation finds the difference between the true sphere radius (in this case one) and the largest eigenvalue of the product of the adjustment matrix and the distortion matrix. This is equivalent to finding the difference between the longest semi-axis and the true radius. This term is important in addition to roundness to identify if the size of the sphere is approximately correct. The results are included in Figure 15 in the right column. As with the roundness, the optimal value is zero.

Here it is noteworthy that for very small amounts of data, the algorithm that finds the matrix $\mathbf{E}$ does a good job approximating the radius. This is not particularly helpful, though, when the roundness and the bias estimation are poor. It is also somewhat interesting to note that the radius error does not seem to change very much as the number of samples increases.

6.8.5 Path Selection

It is also useful to compare the behavior of the convergence given different test points. This gives some concept of how well the training is performed under different training methods. There were three paths selected. The random-points data set is useful because it is easier to analyze and the data points are simple to generate. The



(a) $\hat{\mathbf{A}}$ Roundness (b) $\hat{\mathbf{A}}$ Radius

(c) $\mathbf{E}$ Roundness (d) $\mathbf{E}$ Radius

(e) $\hat{\mathbf{E}}$ Roundness (f) $\hat{\mathbf{E}}$ Radius

Figure 15. Estimates of the roundness (left) and the radius (right) for each technique (top to bottom.) Each line represents a different standard deviation of noise as identified in the legend. Roundness is calculated as the difference between the largest and smallest modulo of the eigenvalues calculated from the product of the adjustment matrix ($\hat{\mathbf{A}}, \sqrt{\mathbf{E}}, \hat{\mathbf{E}}$), and the distortion matrix ($\mathbf{K}$). The optimal value is zero. The radius is the difference of the maximum eigenvalue of the same matrix calculation from the true radius of 1. The optimal value is zero. Experiment used a random walk simulation.

selection of axial turns was chosen because it was included in the existing literature [12]. The third is a random walk technique meant to simulate the movements that would be performed by a person trying to train the headset. Random walks were also used in the literature, though the details of the creation of the random path was not described. Another strategy was listed using a collection of six static positions, but it failed to give good results in prior work and so it was not investigated in this work.

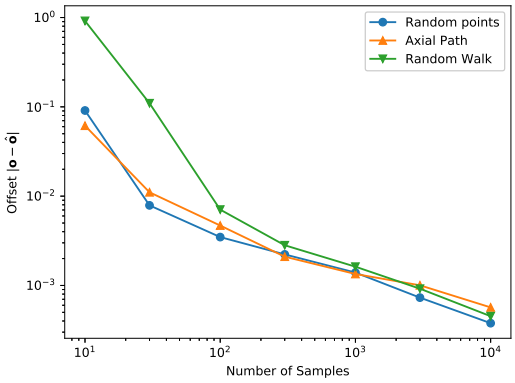
Results of experiments testing for path selection is included in Figure 16. This simulation was run over all algorithms and uses a specific noise standard deviation of $\sigma = 0.01$.

From the results, it can be seen that the ability for the data to find the correct offset improves with all paths and is quite comparable. The random walk is worse than the other methods for small amounts of data since the paths likely do not represent a full exploration of the space. This is expected, since the data points are not expected to be a good representation of the data. Random processes tend to give the best estimation.

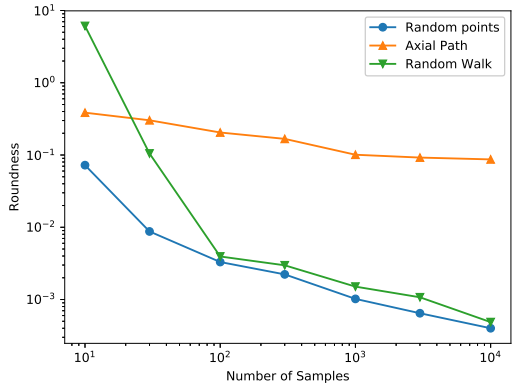
It can be seen, however, that for the roundness and the radius estimation, the axial path is a poor choice. The reason for this is because there is a class of ellipsoids that all represent a good approximation fitting the data if the ellipsoid has a semi-axis in the direction $(1, 1, 0)$. This can be corrected by adding another set of data around the equator. This is a useful indicator that the choice of the data influences the successful estimate for calibration.

It also appears that the roundness and radius are poorly estimated for small amounts of data for the random walk method. This can be explained by the offset being poorly selected which leads to a convergence that may well represent the available data but is a poor global estimate due to the localized nature of short random walks.

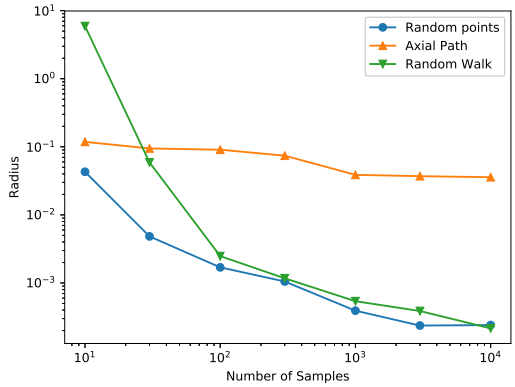
Once there are more than 100 data points, however, the random walk performs



(a) $\hat{\mathbf{A}}$



(b) $\mathbf{E}$



(c) $\hat{\mathbf{E}}$

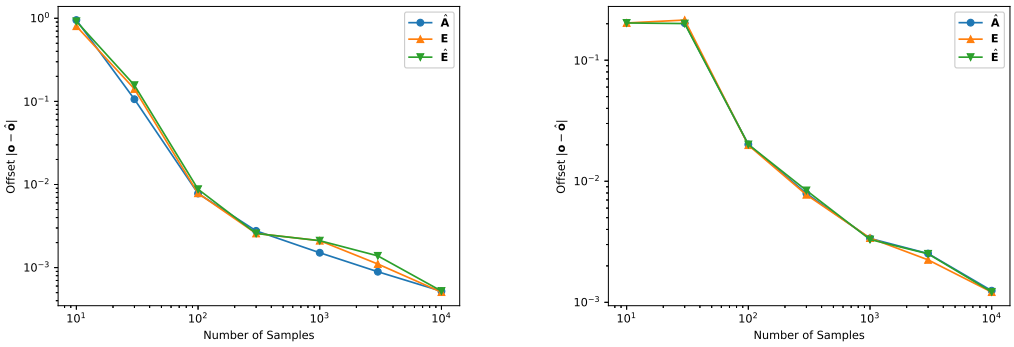
Figure 16. Comparison of the metrics as a function of the nature of the path. These results include results using all three optimization techniques. The noise for these samples had a standard deviation $\sigma = 0.01$. Optimal values are zero for all three metrics.

approximately as well as a set of random points across all metrics. This indicates that having the user rotate the headset around randomly is a good strategy and that it is likely better than trying to move through some specific set of rotation points.

6.8.6 Method Comparison

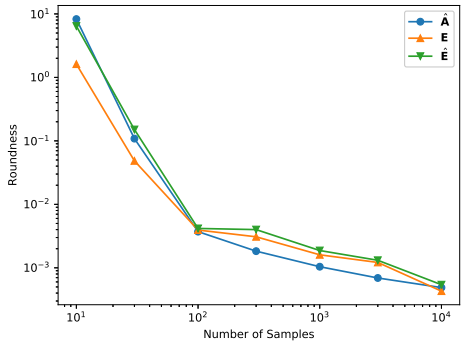
To compare performance of the methods, the random walk method was used for the data selection. Two different noise parameters is used, $\sigma = 0.01$ and $\sigma = 0.0316$ as these values are closest to realistic values. Results are found in Figure 17.

It seems that all three methods provide very similar results across the three different algorithms. This is especially true in the estimation of the offset. It seems that the $\hat{A}$ version may have a small advantage as the number of samples increases, especially for the radius estimate.

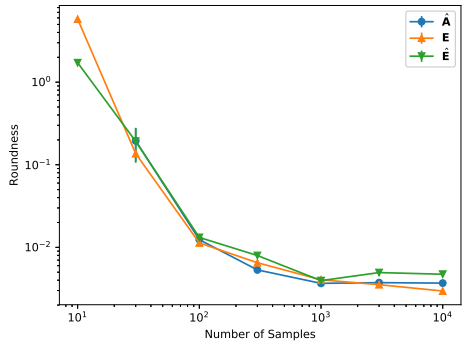


(a) Offset for $\sigma = 0.01$

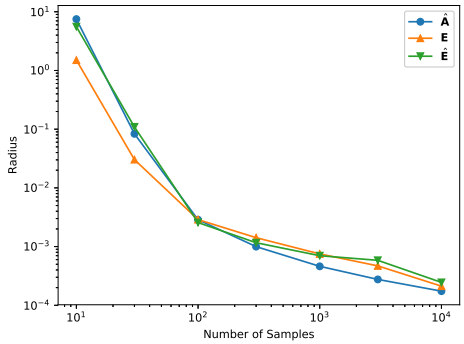
(b) Offset for $\sigma = 0.0316$



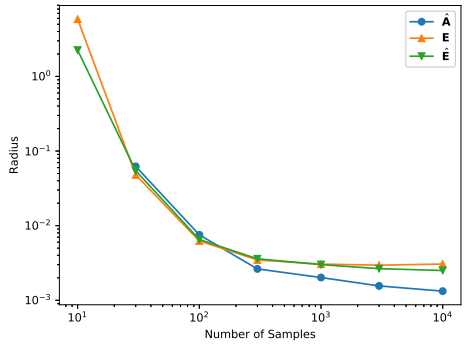
(c) Roundness for $\sigma = 0.01$



(d) Roundness for $\sigma = 0.0316$



(e) Radius for $\sigma = 0.01$



(f) Radius for $\sigma = 0.0316$

Figure 17. These diagrams depict the metrics as run across the three different methods. Figures on the left represent measurements for a standard deviation of noise of $\sigma = 0.01$. The figures on the right are for a standard deviation of $\sigma = 0.0316$.

CHAPTER VII

ORIENTATION

7.1 Aim

From the prior stage, the sensors should all be optimized to give mutually-orthogonal measurements of the B-field scaled by the B-field measured during the training process.

It is assumed that the sensors are all measuring the same B-field and that there are no local differences between the B-field being measured at each of the sensors. The calibration step is under some arbitrary rotation, but to be able to integrate the information from the different sensors, it is necessary to place all the sensors into a common alignment.

In this chapter, $\mathbf{b}_c$ represents the calibrated training values from the prior section, again taken in the absence of a tracking magnet. In this section, k is used to index samples and i will be used to index the sensors. This section outlines a method to find $\mathbf{R}_i$ to find rotations to get all the data into the same orientation space.

In the context of the physical system, the sensor on the top of the user's head located far from the tracking sensors is the reference sensor. The calibrated measurements from the reference sensor is denoted as $\mathbf{b}_{\text{ref}}[k]$. It is assumed that each of the sensors is sampled together and that the values of k correspond between the sensors.

7.2 Quaternions

When trying to perform rotations, using quaternions provides useful properties. Quaternions will also be used more in subsequent stages of this system as well. Because the

concept may be unfamiliar, a brief overview of quaternion math is presented, in particular as they can be used to define rotation in a three-dimensional space.

Quaternions have three vector elements that match the following properties.

$$\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1 \tag{56}$$

The set of quaternions is typically denoted $\mathbb{H}$, a reference to William Rowan Hamilton, the one who first described quaternion algebra.

The quaternions are a non-commutative field. Unit quaternions are used for performing rotations; rotations are non-commutative, so this property is expected.

To simplify notation, given a quaternion of the form $\mathbf{q} = q_w + q_x\mathbf{i} + q_y\mathbf{j} + q_z\mathbf{k}$ is denoted as $\mathbf{q} = \langle q_w, q_x, q_y, q_z \rangle$. Much like complex numbers, the quaternion conjugate is given by $\mathbf{q}^* = \langle q_w, -q_x, -q_y, -q_z \rangle$.

Quaternions of the form $q_x\mathbf{i}+q_y\mathbf{j}+q_z\mathbf{k}$ are called pure quaternions, a space denoted $\mathbb{H}_p$.

To rotate vectors by an angle ψ around a unit vector $\mathbf{u}$, the rotation quaternion is expressed by the following equation.

$$\begin{aligned} q &= \left\langle \cos\left(\frac{\psi}{2}\right), \sin\left(\frac{\psi}{2}\right)\mathbf{u} \right\rangle \\ &= \left\langle \cos\left(\frac{\psi}{2}\right), \sin\left(\frac{\psi}{2}\right)u_x, \sin\left(\frac{\psi}{2}\right)u_y, \sin\left(\frac{\psi}{2}\right)u_z \right\rangle. \end{aligned} \tag{57}$$

The rotation of a quaternion vector $\mathbf{p}$ is given by the expression

$$\mathbf{p}_{\text{rot}} = \mathbf{qpq}^*. \tag{58}$$

The quaternion representation of a vector $\mathbf{v}$ is given by $\langle 0, v_x, v_y, v_z \rangle$.

It is also possible to convert this rotation to a corresponding rotation matrix by the following formula.

$$\mathbf{Q} = \begin{bmatrix} 1-2(q_y^2+q_z^2) & 2(q_xq_y-q_wq_z) & 2(q_xq_z+q_wq_y) \\ 2(q_xq_y+q_wq_z) & 1-2(q_x^2+q_z^2) & 2(q_yq_z-q_wq_x) \\ 2(q_xq_z-q_wq_y) & 2(q_yq_z+q_wq_x) & 1-2(q_x^2+q_y^2) \end{bmatrix} \tag{59}$$

It is also possible to go the other way and generate a quaternion from a rotation matrix, provided that $\mathrm{Tr}\,(\mathbf{Q}) > -1$ [68].

$$q_w = \pm\frac{\sqrt{1+Q_{1,1}+Q_{2,2}+Q_{3,3}}}{2} \tag{60}$$

$$q_x = \frac{Q_{3,2}-Q_{2,3}}{4q_w} \tag{61}$$

$$q_y = \frac{Q_{1,3}-Q_{3,1}}{4q_w} \tag{62}$$

$$q_z = \frac{Q_{2,1}-Q_{1,2}}{4q_w} \tag{63}$$

In practice, if $\mathrm{Tr}\,(\mathbf{M}) = -1+\epsilon$ for ϵ small is numerically unstable. It is also possible that $q_w \approx 0$, which makes the calculations numerically unreliable. There are equivalent methods to find the quaternion from the matrix $\mathbf{Q}$. The stability can be evaluated by simple addition expressions of sums and differences of the diagonal of the matrix $\mathbf{Q}$ to determine if the division will be numerically stable and an appropriate method can be selected that will be numerically stable.

$$q_x = \pm\frac{\sqrt{1+Q_{1,1}-Q_{2,2}-Q_{3,3}}}{2} \tag{64}$$

$$q_w = \frac{Q_{3,2}-Q_{2,3}}{4q_x} \tag{65}$$

$$q_y = \frac{Q_{1,2}+Q_{2,1}}{4q_x} \tag{66}$$

$$q_z = \frac{Q_{1,3}+Q_{3,1}}{4q_x} \tag{67}$$

$$q_y = \pm\frac{\sqrt{1 + Q_{1,1} + Q_{2,2} - Q_{3,3}}}{2} \tag{68}$$

$$q_w = \frac{Q_{1,3} - Q_{3,1}}{4q_y} \tag{69}$$

$$q_x = \frac{Q_{1,2} + Q_{2,1}}{4q_y} \tag{70}$$

$$q_z = \frac{Q_{2,3} + Q_{3,2}}{4q_y} \tag{71}$$

$$q_z = \pm\frac{\sqrt{1 - Q_{1,1} - Q_{2,2} + Q_{3,3}}}{2} \tag{72}$$

$$q_w = \frac{Q_{2,1} - Q_{1,2}}{4q_z} \tag{73}$$

$$q_x = \frac{Q_{1,3} + Q_{3,1}}{4q_z} \tag{74}$$

$$q_y = \frac{Q_{2,3} + Q_{3,2}}{4q_z} \tag{75}$$

The group of 3D rotations about the origin in three-dimensional Euclidean space $\mathbb{R}^3$ is mathematically titled the special orthogonal group SO(3) under matrix multiplication. Special matrices are matrices that are invertible with determinant one. Orthogonal matrices have column vectors that are orthonormal [30]. This forms a compact Lie group having skew-symmetric matrices as its Lie algebra. The Lie algebra is a non-associative vector space with the binary operation

$$[\mathbf{A}, \mathbf{B}] = \mathbf{AB} - \mathbf{BA} \tag{76}$$

This is known as the Lie bracket operation [33]. The properties of the Lie bracket are outlined. In these calculations, let $\mathcal{G}$ reprsent the vector space over some field $\mathbf{F}$ with the Lie bracket operation $[\cdot, \cdot] : \mathcal{G} \times \mathcal{G} \to \mathcal{G}$. Let $\mathbf{A}, \mathbf{B}, \mathbf{C} \in \mathcal{G}$, $x, y \in F$.

- Bilinearity

$$[x\mathbf{A} + y\mathbf{B}, \mathbf{C}] = x[\mathbf{A}, \mathbf{C}] + y[\mathbf{B}, \mathbf{C}] \tag{77}$$

$$[\mathbf{C}, x\mathbf{A} + y\mathbf{B}] = x[\mathbf{C}, \mathbf{A}] + y[\mathbf{C}, \mathbf{B}] \tag{78}$$

- Alternativity

$$[\mathbf{A}, \mathbf{A}] = 0 \tag{79}$$

- Jacobi identity

$$[\mathbf{A}, [\mathbf{B}, \mathbf{C}]] + [\mathbf{C}, [\mathbf{A}, \mathbf{B}]] + [\mathbf{B}, [\mathbf{C}, \mathbf{A}]] = 0 \tag{80}$$

This set of properties describe the properties that are exhibited by the matrices of rotations and outlines the operations that can be applied to them. Furthermore, results of Lie analysis identify that the Lie algebra SO(3) is isomorphic to the Lie algebra $\mathbb{R}^3$ with the field $\mathbb{R}$ and the Lie bracket operation of cross product, which means that the underlying dimensionality of the space is 3. It happens that unit quaternions are a valid representation of SO(3) where the imaginary part can be viewed as three values describing the rotation (as the fourth can be calculated by the unitary nature of the vector.)

The Rodrigues rotation formula is an algorithm for rotating a vector in $\mathbb{R}^3$ given a unit vector of rotation $\mathbf{u}$ and angle of rotation θ.

$$\mathbf{y} = \cos\theta\mathbf{x} + \sin\theta(\mathbf{u} \times \mathbf{x}) + (1 - \cos\theta)\mathbf{u}(\mathbf{u} \cdot \mathbf{x}) \tag{81}$$

Olinde Rodrigues was a contemporary of Hamilton, the inventor of quaternions, while researching homogeneous transformations. A rotation matrix can be found using the Rodrigues rotation formula given by:

$$\mathbf{R} = \cos\theta\mathbf{I} + \sin\theta[\mathbf{u}]_\times + (1 - \cos\theta)\mathbf{u}\mathbf{u}^\intercal. \tag{82}$$

Here, $[\mathbf{u}]_\times$ is the matrix such that $[u]_\times \mathbf{b} = \mathbf{u} \times \mathbf{b}$ for all $\mathbf{b} \in \mathbb{R}^3$. This is accomplished by the skew-symmetric matrix below.

$$[\mathbf{u}]_\times = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix} \tag{83}$$

In the analysis, often it is beneficial to perform calculations in matrices to allow the use of tools from matrix calculus. Using quaternions for rotations is beneficial because they can be expressed by three values (since quaternion rotations are unit vectors) rather than the nine required for a matrix. For performing rotations on vectors, matrices also are more computationally efficient. However, performing a sequence of rotations is done more efficiently in quaternions and quaternion vectors are simpler to intuitively connect to the rotation parameters. Quaternions also provide a set of additional tools for performing a variety of calculations efficiently.

The unit quaternions are isomorphic to the special unitary group $\mathrm{SU}(2) \subset \mathbb{C}^{2\times 2}$ and given a quaternion $\mathbf{q}$, it can be used to generate a 2×2 unitary matrix.

$$\begin{bmatrix} q_0 + iq_1 & q_2 + iq_3 \\ -q_2 + iq_3 & q_0 - iq_1 \end{bmatrix} \tag{84}$$

There is a bijection from SU(2) to SO(3), but it is a double covering. This can be demonstrated by recognizing $\mathbf{q}$ and $-\mathbf{q}$ represent the same rotation. The easiest way to show this is that $(-\mathbf{q})\mathbf{p}(-\mathbf{q}^*) = \mathbf{q}\mathbf{p}\mathbf{q}^*$. Another way to look at this is to recognize that changing the sign on the first term gives $-cos(x) = cos(x + \pi)$ and for the imaginary terms, $-sin(x) = sin(-x) = -sin(-x + \pi)$. Because the argument to the sinusoidal functions is half the angle of rotation, $x = \frac{\psi}{2}$, $-\mathbf{q}$ represents rotating the negative angle in the opposite rotation direction (i.e. the same rotation), then rotating an additional full turn, making no change to the final rotation position. Thus, $\mathbf{q}$ and $-\mathbf{q}$ describe the same rotation.

For purposes of optimizations, it is beneficial to be able to use unconstrained optimizations. To accomplish this, rather than using quaternions directly in the optimization, instead the values can be parameterized by (ψ, θ, ϕ), where θ and ϕ are the angular spherical coordinate terms for the unit vector of rotation. The rotation about that unit vector is denoted by ψ. These values are periodic, which must be considered, but it is unnecessary to specify boundary conditions; this is also a double mapping of SO(3) because $\{\psi, \theta, \phi\} \sim \{-\psi, \pi - \theta, \pi + \phi\}$ where $\sim$ represents an equivalence relation in SO(3) (i.e. two representations of the same rotation.)

7.3 Optimizations

Three different optimizations were done to solve this problem. Each was done using Nelder-Mead and BFGS quasi-Newton methods. Three different ways of measuring the fitness of a rotation were explored. These are depicted in Figure 18. The first objective function is the one based on the great-circle-distance, representing the shortest path along the surface of a sphere between two points on the sphere. This is equivalent to finding the corresponding angle θ. The second technique uses the Euclidean distance between the two measurements. The third uses the Euclidean distance between points projected to the surface of the sphere.

7.3.1 Central Angle Distance Rotation

This first method aims to find the rotation matrix $\mathbf{Q}$ which minimizes the L^2 norm of the vector of central angle distances between the test sensor measurements and the reference sensor measurements. This is described by the following objective function.

$$f_{\text{CAD}}(\mathbf{Q}) = \sum_k \left[\arccos \frac{\mathbf{b}_{\text{ref}}[k]^\intercal \mathbf{Q} \mathbf{b}_{\text{c}}[k]}{\|\mathbf{b}_{\text{ref}}[k]\| \|\mathbf{Q}\mathbf{b}_{\text{c}}[k]\|} \right]^2 \tag{85}$$

In practice, the rotation is performed using quaternions, though calculus over quaternions is an active area of research and presents a special set of challenges analytically. For the purposes of this optimization, it is useful to mathematically analyze

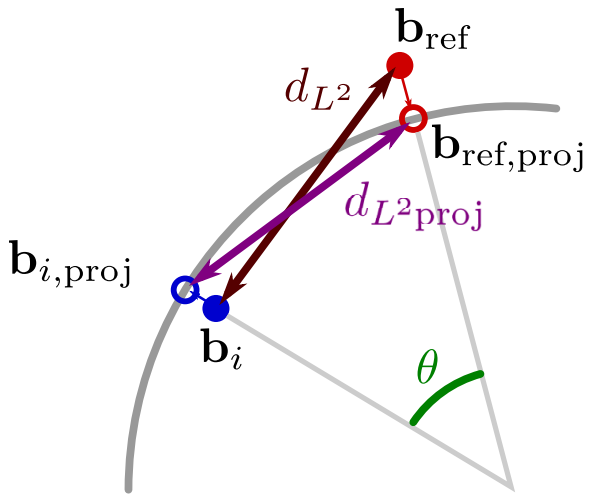


Figure 18. Illustration of three different ways of mesauring rotation distances. The solid dots represent two different measurements with the distance between them represented by d_{L^2}. The circles represent a projection of these values onto a unit circle, the distance between the projections given by $d_{L^2\text{proj}}$. The third method finds the smallest angle θ between the two measurements.

the problem using matrix calculus and then doing implementation with quaternions. Even in this framework, the calculation of the Jacobian takes a bit of work.

It is also useful to recognize that $\|\mathbf{Q}\mathbf{b}_c[k]\| = \|\mathbf{b}_c[k]\|$ because rotation matrices are L^2-norm-preserving operators.

To prove this, first recognize that the L^2 norm can be calculated as $\|\mathbf{b}\| = \sqrt{\mathbf{b}^\intercal\mathbf{b}}$. The L^2 norm of the rotated vector is given by $\|\mathbf{Q}\mathbf{b}\| = \sqrt{\mathbf{b}^\intercal\mathbf{Q}^\intercal\mathbf{Q}\mathbf{b}}$. Because Q is an orthogonal matrix, $\mathbf{Q}^\intercal\mathbf{Q} = \mathbf{I}$, so $\|\mathbf{Q}\mathbf{b}\| = \sqrt{\mathbf{b}^\intercal\mathbf{b}} = \|\mathbf{b}\|$.

For notation, let the dot product of the unit vectors be described by $\xi[k]$.

$$\xi[k] = \frac{\mathbf{b}_{\text{ref}}[k]^\intercal\mathbf{Q}\mathbf{b}_c[k]}{\|\mathbf{b}_{\text{ref}}[k]\|\|\mathbf{Q}\mathbf{b}_c[k]\|} \tag{86}$$

Then the Jacobian with respect to the matrix $\mathbf{Q}$ is specified by the following calculation.

$$\frac{\partial f_{\text{CAD}}(\mathbf{Q})}{\partial\mathbf{Q}} = -2\sum_k \frac{\arccos\xi[k]}{\sqrt{1-\xi^2[k]}}\frac{\partial\xi[k]}{\partial\mathbf{Q}} \tag{87}$$

$$\frac{\partial \xi[k]}{\partial \mathbf{Q}} = \frac{\mathbf{b}_{\text{ref}}[k]\mathbf{b}_{\text{c}}[k]^{\intercal}}{\|\mathbf{b}_{\text{ref}}[k]\|\|\mathbf{Q}\mathbf{b}_{\text{c}}[k]\|} \tag{88}$$

A weakness of this Jacobian is that it includes a numerically unstable calculation.

$$f(x) = -\frac{\arccos(\xi)}{\sqrt{1-\xi^2}} \tag{89}$$

This function has singularities at $x = -1$ and $x = 1$. The first singularity is a pole where $\lim_{x\to -1^+} f(x) = -\infty$. The instability at this position is fine as it grows without bound. The second singularity is more complicated. First, recognize that the function is a removable singularity by the Riemann Removable Singularities Theorem [39].

$$\lim_{x\to 1^-} f(x) = -1 \tag{90}$$

The function is numerically unstable near this point. This is of particular concern because this position represents when the two vectors are maximally similar, which is the targeted goal of the optimization. This may lead to a failure of convergence when the vectors become very similar leading to a value of $\xi = 1$.

This function represents a real-valued continuous function defined on $(-1, 1] \in \mathbb{R}$ that is differentiable on $(0, 1)$. A Taylor series approximation is used to remove the instability.

Taylor's Theorem:[22] Let n be a non-negative integer and suppose that f is a real-valued function defined and continuous on the closed interval $[a, b]$ of $\mathbb{R}$, such that the derivatives of f of order up to and including n are defined and continuous on $[a, b]$, $f^{(n)}$ is differentiable on the open interval (a, b) and $f^{(n+1)}$ is integrable on (a, b). Then, for each $x \in [a, b]$,

$$f(x) = f(a) + (x-a)f'(a) + \cdots + \frac{(x-a)^n}{n!}f^{(n)}(a) + \int_a^x \frac{(x-t)^n}{n!}f^{(n+1)}(t)\mathrm{d}t. \tag{91}$$

Because the equation is being evaluated at a specific value of $\xi \in [-1, 1]$, it must only be shown that f from Equation 89 is continuous on $[-1, 1]$, n-times differentiable on $(0, 1)$, and $f^{(n+1)}$ is integrable on (a, b).

First step is to prove continuity on $[-1, 1]$. For $f, g \in \mathcal{C}$, $\frac{f}{g} \in \mathcal{C}$ whenever $g \neq 0$. From this, there is continuity on $(0, 1)$ because the numerator and denominator are continuous functions. It remains to show continuity on $\{-1, 1\}$. As was already identified, the limit $\lim_{\xi \to 1^-} f(\xi) = 1$ by L'Hôpital's Rule, since the numerator and denominator are both differentiable in the left-hand neighborhood around 1. Another way to look at it, $\sqrt{1-\xi^2} < \arccos(\xi) < (2-\xi)\sqrt{1-\xi^2}$ on the region $[0, 1)$ and thus $1 < f(\xi) < (2 - \xi)$ for $\xi \in [0, 1)$, so the function is continuous from the left.

The function is also continuous to the right as it approaches 1, but it converges to ∞. The function is monotonically increasing to the right.

The function has derivatives of order f up to and including n where here $n = 5$. In fact, assuming that $\arccos(x) \in \mathcal{C}^k$, then $f(\xi) \in \mathcal{C}^{k-1}$ because

$$f(\xi) = \arccos(\xi) \frac{\mathrm{d} \arccos(\xi)}{\mathrm{d}\xi} \tag{92}$$

The consequence follows by sequential applications of the product rule.

The function is integrable, but the importance of the integration term is to bound the error. It can be easily recognized that the error grows without bound as the function approaches -1 because $f(x)$ is unbounded whereas the Taylor series expansion is bounded on the interval. This point represents a complete misalignment of the vectors and is unlikely to occur in the optimization, especially with a method for finding an initial guess, and the function is decreasing on the region so using the Taylor approximation mostly serves to slow down the optimization. In practice, the Taylor series expansion should not be used over that region since the function itself is numerically stable and the Taylor series becomes a poor approximation close to -1.

With the considerations of the Taylor series met, a Taylor series expansion is

Table 1. Table of coefficients for the Taylor Series expansion about the points x=0.5 and x=0.9. These coefficients account for the factorial denominators from the standard Taylor series form.

k	0	1	2	3	4	5
$\alpha_k(a = 0.5)$	1.2092	-0.5272	0.278933	-0.158697	0.0937861	-0.0567337
$\alpha_k(a = 0.9)$	1.03473	-0.361822	0.152124	-0.068572	0.032064	-0.015336

performed about the point $a = 0.5$ and $a = 0.9$.

$$\frac{\arccos(x)}{\sqrt{1-x^2}} \approx \sum_{k=0}^{n} \alpha_k \left(x-a\right)^k \tag{93}$$

In the implementation, $n = 5$ and coefficients for the provided points are specified in Table 1.

This function is a poor representation of the equation near -1 as the Taylor expansion is bounded unlike the original. This is not a major concern, however, since the monotonicity of the function is preserved so the gradient descent will still converge, albeit more slowly, for highly dissimilar vectors. Rather than using the approximation at all points, however, the normal calculation method is used on $[-1, a]$ and the Taylor series estimation is used on $[a, 1]$. The estimate is quite accurate over that region. This serves to remove the numerical instability near 1.

It is also possible to use a Padé approximant [24], though the region where the application on $[a, 1]$ has well-behaved convergence, so it is unnecessary.

This gradient is a matrix with the same dimensionality as $\mathbf{Q}$. To find the relative Jacobian with respect to the rotational terms ψ, θ, ϕ, a matrix form of the chain rule is used. In matrix calculus, the chain rule when the derivative of a scalar is taken by means of a matrix, a trace function is used.

$$\frac{\partial f_{\text{CAD}}}{\partial \psi} = \text{Tr}\, \frac{\partial f_{\text{CAD}}}{\mathbf{Q}}^{\intercal} \frac{\partial \mathbf{Q}}{\partial \psi} \tag{94}$$

$$\frac{\partial f_{\text{CAD}}}{\partial \theta} = \text{Tr}\, \frac{\partial f_{\text{CAD}}}{\mathbf{Q}}^{\intercal} \frac{\partial \mathbf{Q}}{\partial \theta} \tag{95}$$

$$\frac{\partial f_{\text{CAD}}}{\partial \phi} = \text{Tr}\, \frac{\partial f_{\text{CAD}}}{\mathbf{Q}}^{\intercal} \frac{\partial \mathbf{Q}}{\partial \phi} \tag{96}$$

This requires the calculation of the partial derivatives with $\mathbf{Q}$ with respect to each of the rotational values. Fortunately, these matrices must only be calculated one time for each iteration of the optimization as it is dependent upon $\mathbf{Q}$ and it is independent of the iteration number k, and it can be factored out to only require one matrix multiplication for each of the terms ψ, θ, ϕ if the full matrix $^{\partial f_{\text{CAD}}}/_{\partial \mathbf{Q}}$ is preserved over each k.

The expressions for the partial derivatives are fairly complex; they will be reused for other optimizations and are included here as a reference.

$$\mathbf{Q}(\psi,\theta,\phi) = \begin{bmatrix} Q_{1,1} & Q_{1,2} & Q_{1,3} \\ Q_{2,1} & Q_{2,2} & Q_{2,3} \\ Q_{3,1} & Q_{3,2} & Q_{3,3} \end{bmatrix} \tag{97}$$

$$Q_{1,1} = 1 - (1 - \cos\psi)(\sin^2\theta \sin^2\phi + \cos^2\theta) \tag{97a}$$

$$Q_{1,2} = \frac{1}{2}(1 - \cos\psi)\sin^2\theta \sin 2\phi - \sin\psi\cos\theta \tag{97b}$$

$$Q_{1,3} = \frac{1}{2}(1 - \cos\psi)\sin 2\theta \cos\phi + \sin\psi \sin\theta \sin\phi \tag{97c}$$

$$Q_{2,1} = \frac{1}{2}(1 - \cos\psi)\sin^2\theta \sin 2\phi + \sin\psi\cos\theta \tag{97d}$$

$$Q_{2,2} = 1 - (1 - \cos\psi)(\sin^2\theta \cos^2\phi + \cos^2\theta) \tag{97e}$$

$$Q_{2,3} = \frac{1}{2}(1 - \cos\psi)\sin 2\theta \sin\phi - \sin\psi \sin\theta \cos\phi \tag{97f}$$

$$Q_{3,1} = \frac{1}{2}(1 - \cos\psi)\sin 2\theta \cos\phi - \sin\psi \sin\theta \sin\phi \tag{97g}$$

$$Q_{3,2} = \frac{1}{2}(1 - \cos\psi)\sin 2\theta \sin\phi + \sin\psi \sin\theta \cos\phi \tag{97h}$$

$$Q_{3,3} = 1 - (1 - \cos\psi)\sin^2\theta \tag{97i}$$

$$\frac{\partial \mathbf{Q}(\psi,\theta,\phi)}{\partial \psi} = \begin{bmatrix} \frac{\partial Q}{\partial \psi}_{1,1} & \frac{\partial Q}{\partial \psi}_{1,2} & \frac{\partial Q}{\partial \psi}_{1,3} \\ \frac{\partial Q}{\partial \psi}_{2,1} & \frac{\partial Q}{\partial \psi}_{2,2} & \frac{\partial Q}{\partial \psi}_{2,3} \\ \frac{\partial Q}{\partial \psi}_{3,1} & \frac{\partial Q}{\partial \psi}_{3,2} & \frac{\partial Q}{\partial \psi}_{3,3} \end{bmatrix} \tag{98}$$

$$\frac{\partial Q}{\partial \psi}_{1,1} = -\sin\psi(\sin^2\theta\sin^2\phi + \cos^2\theta) \tag{98a}$$

$$\frac{\partial Q}{\partial \psi}_{1,2} = \frac{1}{2}\sin\psi\sin^2\theta\sin 2\phi - \cos\psi\cos\theta \tag{98b}$$

$$\frac{\partial Q}{\partial \psi}_{1,3} = \frac{1}{2}\sin\psi\sin 2\theta\cos\phi + \cos\psi\sin\theta\sin\phi \tag{98c}$$

$$\frac{\partial Q}{\partial \psi}_{2,1} = \frac{1}{2}\sin\psi\sin^2\theta\sin 2\phi + \cos\psi\cos\theta \tag{98d}$$

$$\frac{\partial Q}{\partial \psi}_{2,2} = -\sin\psi(\sin^2\theta\cos^2\phi + \cos^2\theta) \tag{98e}$$

$$\frac{\partial Q}{\partial \psi}_{2,3} = \frac{1}{2}\sin\psi\sin 2\theta\sin\phi - \cos\psi\sin\theta\cos\phi \tag{98f}$$

$$\frac{\partial Q}{\partial \psi}_{3,1} = \frac{1}{2}\sin\psi\sin 2\theta\cos\phi - \cos\psi\sin\theta\sin\phi \tag{98g}$$

$$\frac{\partial Q}{\partial \psi}_{3,2} = \frac{1}{2}\sin\psi\sin 2\theta\sin\phi + \cos\psi\sin\theta\cos\phi \tag{98h}$$

$$\frac{\partial Q}{\partial \psi}_{3,3} = -\sin\psi\sin^2\theta \tag{98i}$$

$$\frac{\partial \mathbf{Q}(\psi,\theta,\phi)}{\partial \theta} = \begin{bmatrix} \frac{\partial Q}{\partial \theta}_{1,1} & \frac{\partial Q}{\partial \theta}_{1,2} & \frac{\partial Q}{\partial \theta}_{1,3} \\ \frac{\partial Q}{\partial \theta}_{2,1} & \frac{\partial Q}{\partial \theta}_{2,2} & \frac{\partial Q}{\partial \theta}_{2,3} \\ \frac{\partial Q}{\partial \theta}_{3,1} & \frac{\partial Q}{\partial \theta}_{3,2} & \frac{\partial Q}{\partial \theta}_{3,3} \end{bmatrix} \tag{99}$$

$$\frac{\partial Q}{\partial \theta}_{1,1} = 2\sin^2\frac{\psi}{2}\sin 2\theta\cos^2\phi \tag{99a}$$

$$\frac{\partial Q}{\partial \theta}_{1,2} = \sin^2\frac{\psi}{2}\sin 2\theta\sin 2\phi + \sin\psi\sin\theta \tag{99b}$$

$$\frac{\partial Q}{\partial \theta}_{1,3} = 2\sin^2\frac{\psi}{2}\cos 2\theta\cos\phi + \sin\psi\cos\theta\sin\phi \tag{99c}$$

$$\frac{\partial Q}{\partial \theta}_{2,1} = \sin^2\frac{\psi}{2}\sin 2\theta\sin 2\phi - \sin\psi\sin\theta \tag{99d}$$

$$\frac{\partial Q}{\partial \theta}_{2,2} = 2\sin^2\frac{\psi}{2}\sin 2\theta\sin^2\phi \tag{99e}$$

$$\frac{\partial Q}{\partial \theta}_{2,3} = 2\sin^2\frac{\psi}{2}\cos 2\theta\sin\phi - \sin\psi\cos\theta\cos\phi \tag{99f}$$

$$\frac{\partial Q}{\partial \theta}_{3,1} = 2\sin^2\frac{\psi}{2}\cos 2\theta\cos\phi - \sin\psi\cos\theta\sin\phi \tag{99g}$$

$$\frac{\partial Q}{\partial \theta}_{3,2} = 2\sin^2\frac{\psi}{2}\cos 2\theta\sin\phi + \sin\psi\cos\theta\cos\phi \tag{99h}$$

$$\frac{\partial Q}{\partial \theta}_{3,3} = -2\sin^2\frac{\psi}{2}\sin 2\theta \tag{99i}$$

$$\frac{\partial \mathbf{Q}(\psi,\theta,\phi)}{\partial \phi} = \begin{bmatrix} \frac{\partial Q}{\partial \phi}_{1,1} & \frac{\partial Q}{\partial \phi}_{1,2} & \frac{\partial Q}{\partial \phi}_{1,3} \\ \frac{\partial Q}{\partial \phi}_{2,1} & \frac{\partial Q}{\partial \phi}_{2,2} & \frac{\partial Q}{\partial \phi}_{2,3} \\ \frac{\partial Q}{\partial \phi}_{3,1} & \frac{\partial Q}{\partial \phi}_{3,2} & \frac{\partial Q}{\partial \phi}_{3,3} \end{bmatrix} \tag{100}$$

$$\frac{\partial Q}{\partial \phi}_{1,1} = -2\sin^2\frac{\psi}{2}\sin^2\theta\sin 2\phi \tag{100a}$$

$$\frac{\partial Q}{\partial \phi}_{1,2} = 2\sin^2\frac{\psi}{2}\sin^2\theta\cos 2\phi \tag{100b}$$

$$\frac{\partial Q}{\partial \phi}_{1,3} = -\sin^2\frac{\psi}{2}\sin 2\theta\sin\phi + \sin\psi\sin\theta\cos\phi \tag{100c}$$

$$\frac{\partial Q}{\partial \phi}_{2,1} = 2\sin^2\frac{\psi}{2}\sin^2\theta\cos 2\phi \tag{100d}$$

$$\frac{\partial Q}{\partial \phi}_{2,2} = 2\sin^2\frac{\psi}{2}\sin^2\theta\sin 2\phi \tag{100e}$$

$$\frac{\partial Q}{\partial \phi}_{2,3} = \sin^2\frac{\psi}{2}\sin 2\theta\cos\phi + \sin\psi\sin\theta\sin\phi \tag{100f}$$

$$\frac{\partial Q}{\partial \phi}_{3,1} = -\sin^2\frac{\psi}{2}\sin 2\theta\sin\phi - \sin\psi\sin\theta\cos\phi \tag{100g}$$

$$\frac{\partial Q}{\partial \phi}_{3,2} = \sin^2\frac{\psi}{2}\sin 2\theta\cos\phi - \sin\psi\sin\theta\sin\phi \tag{100h}$$

$$\frac{\partial Q}{\partial \phi}_{3,3} = 0 \tag{100i}$$

7.3.2 Euclidean Rotation

This objective function optimizes the Euclidean distance between corresponding field measurements.

$$f_{\mathrm{L2}}(\mathbf{Q}) = \sum_k \left\|\mathbf{b}_{\mathrm{ref}}[k] - \mathbf{Q}\mathbf{b}_{\mathrm{c}}[k]\right\|^2 \tag{101}$$

The Jacobian is calculated using the following function.

$$\frac{\partial f_{\mathrm{L2}}(\mathbf{Q})}{\partial \mathbf{Q}} = 2\left(\mathbf{b}_c[k]\mathbf{b}_c[k]^\intercal\mathbf{Q}^\intercal - \mathbf{b}_c[k]\mathbf{b}_{\mathrm{ref}}^\intercal\right) \tag{102}$$

$$\frac{\partial f_{\mathrm{L2}}}{\partial \psi} = \mathrm{Tr}\,\frac{\partial f_{\mathrm{L2}}}{\mathbf{Q}}\frac{\partial \mathbf{Q}}{\partial \psi} \tag{103}$$

$$\frac{\partial f_{\mathrm{L2}}}{\partial \theta} = \mathrm{Tr}\,\frac{\partial f_{\mathrm{L2}}}{\mathbf{Q}}\frac{\partial \mathbf{Q}}{\partial \theta} \tag{104}$$

$$\frac{\partial f_{\mathrm{L2}}}{\partial \phi} = \mathrm{Tr}\,\frac{\partial f_{\mathrm{L2}}}{\mathbf{Q}}\frac{\partial \mathbf{Q}}{\partial \phi} \tag{105}$$

These equations use Equations eqs. (98) to (100).

7.3.3 Projected Euclidean Rotation

Because the optimization is finding a rotation matrix, an argument could be made that using the Euclidean distance between unit vectors for corresponding samples. This results in nearly the same calculation as in the Euclidean rotation method, but the reference and measurements from the sensor under calibration are normalized. While the calculation is straightforward, the effect is different. The accuracy of this method compared to the normal Euclidean method is dependent on the nature of the noise.

$$f_{\mathrm{PL2}}(\mathbf{Q}) = \sum_k \left\| \frac{\mathbf{b}_{\mathrm{ref}}[k]}{\|\mathbf{b}_{\mathrm{ref}}[k]\|} - \mathbf{Q}\frac{\mathbf{b}_{\mathrm{c}}[k]}{\|\mathbf{b}_{\mathrm{c}}[k]\|} \right\|^2 \tag{106}$$

The Jacobian is calculated using the following function.

$$\frac{\partial f_{\mathrm{PL2}}(\mathbf{Q})}{\partial \mathbf{Q}} = 2\sum_k \left(\frac{\mathbf{b}_c[k]\mathbf{b}_c[k]^{\intercal}\mathbf{Q}^{\intercal}}{\|\mathbf{b}_{\mathrm{c}}[k]\|^2} - \frac{\mathbf{b}_c[k]\mathbf{b}_{\mathrm{ref}}^{\intercal}}{\|\mathbf{b}_{\mathrm{c}}[k]\|\|\mathbf{b}_{\mathrm{ref}}[k]\|} \right) \tag{107}$$

$$\frac{\partial f_{\mathrm{PL2}}}{\partial \psi} = \mathrm{Tr}\,\frac{\partial f_{\mathrm{PL2}}}{\mathbf{Q}}\frac{\partial \mathbf{Q}}{\partial \psi} \tag{108}$$

$$\frac{\partial f_{\mathrm{PL2}}}{\partial \theta} = \mathrm{Tr}\,\frac{\partial f_{\mathrm{PL2}}}{\mathbf{Q}}\frac{\partial \mathbf{Q}}{\partial \theta} \tag{109}$$

$$\frac{\partial f_{\mathrm{PL2}}}{\partial \phi} = \mathrm{Tr}\,\frac{\partial f_{\mathrm{PL2}}}{\mathbf{Q}}\frac{\partial \mathbf{Q}}{\partial \phi} \tag{110}$$

7.3.4 Euclidean Affine Transform

The optimization method here can be extended beyond rotations. It is possible that the sensor calibration optimization may not have been particularly accurate and it may be possible to find a better $\hat{\mathbf{A}}_i, \hat{\mathbf{o}}_i$ to allow a good fit to the reference sensor. This technique is very different from the prior methods as it requires operating on the raw measurement values from the sensor so that the update to $\hat{\mathbf{o}}$ to be applied correctly. Thus this method tries to find a matrix $\tilde{\mathbf{A}}$ and an offset $\tilde{\mathbf{o}}$ to find the optimal affine transformation to transform the raw sensor measurements to the reference sensor estimates.

This technique can theoretically provide a better match to the reference sensor than rotations alone, but there is a potential of overfitting. Rotations have three degrees of freedom whereas the affine transformation has twelve.

$$f_{\text{EAT}} = \sum_k \|\mathbf{b}_{\text{ref}}[k] - \tilde{\mathbf{A}}(\mathbf{b}_i[k] - \tilde{\mathbf{o}})\|^2 \tag{111}$$

The Jacobians with respect to the matrix $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{o}}$ are given by the following expressions.

$$\frac{\partial f_{\text{EAT}}}{\partial \tilde{\mathbf{A}}} = 2 \sum_k \left[\tilde{\mathbf{A}}(\mathbf{b}_i[k] - \tilde{\mathbf{o}}) - \mathbf{b}_{\text{ref}}[k]\right] (\mathbf{b}_i[k] - \tilde{\mathbf{o}})^\intercal \tag{112}$$

$$\frac{\partial f_{\text{EAT}}}{\partial \tilde{\mathbf{o}}} = -2\tilde{\mathbf{A}}^\intercal \sum_k \left[\tilde{\mathbf{A}}(\mathbf{b}_i[k] - \tilde{\mathbf{o}}) - \mathbf{b}_{\text{ref}}[k]\right] \tag{113}$$

7.4 Finding Initial Estimates

Performing the optimization using these functions still follows the same challenges to optimization problems. The rotation problems are periodic, so they are not convex. The rotation optimization is fairly stable for this particular problem, but there is one area of complexity. The nature of the angle ψ, the solution $(0, \theta, \phi)$ represents an

equivalence class of rotations for all θ and ϕ. This means there is a likelihood of the presence of a saddle point. What's more, it is quite possible that there is a non-strict saddle point at that location.

A strict saddle point is a saddle point with a strictly negative curvature in at least one direction. A critical point is one where the derivative (here Jacobian) is zero. Calculating the Laplacian (Hessian) at a point can be used to test the behavior at that point. If the Hessian is positive-definite, the point is an isolated minimum. If Hessian is negative-definite, the point is an isolated maximum. If there are positive and negative eigenvalues, it is a saddle point. Non-strict saddle points occur in the system where at least one eigenvalue is zero and the others are all either non-negative or non-positive.

Conceptually, this represents having a metaphorical valley that is either: increasing or flat *or* decreasing or flat. It could also represent a set of values (such as a line or plane) that contiguously have the same objective function value. This set of values could represent a local maximum, a local minimum, or a non-strict saddle point and distinguishing the difference is an NP-hard problem, as it requires searching to find the behavior of the boundary.

For these problems, however, the plane of constant objective functions is known. If the two sets of values are initially aligned, this set of solutions represents the global minima. Rather than trying to deal with the problems that arise on this plane, it is beneficial to create a method to get an informed initial guess to make an efficient estimate.

7.4.1 Simple Method

One way to do this would be to select a sample index k and find the rotation that moves one point to the other. The estimate of the angle of rotation ψ and the unit vector about which to rotate $\mathbf{v}$ can be found using the following calculations.

$$\psi_0 = \arccos\left(\frac{\mathbf{b}_{\text{ref}}[k]}{\|\mathbf{b}_{\text{ref}}[k]\|} \cdot \frac{\mathbf{b}_{\text{c}}[k]}{\|\mathbf{b}_{\text{c}}[k]\|}\right) \tag{114}$$

$$\mathbf{v}_0 = \frac{\mathbf{b}_{\text{ref}}[k] \times \mathbf{b}_{\text{c}}[k]}{\|\mathbf{b}_{\text{ref}}[k] \times \mathbf{b}_{\text{c}}[k]\|} \tag{115}$$

This assumes that the two vectors are not collinear otherwise the normalization of the vector $\mathbf{v}_0$ will cause division by zero. Of course in that case, the dot product will be $-1, 1$. In the first case, any vector orthogonal to the set of vectors with an angle of rotation of π is a solution. In the second case, no rotation is necessary.

In 3D space, though, this solution is not the only rotation. This represents the geodesic rotation, i.e., the shortest possible rotation. Rather than doing this, another technique was created to attempt to find the rotation that is close to a common rotation between two vector pairs.

7.4.2 Characterization of Viable Rotations

This method finds the estimated best-fit rotation to rotate a pair of vectors onto another pair of vectors. To do this, recognize that the set of all axis vectors to rotate one vector onto another exist on the hemisphere of the unit sphere. This is true because the curve of rotation must be symmetric about a central line and that central line must be orthogonal to the difference of the vectors $\hat{\mathbf{y}} - \hat{\mathbf{x}}$.

This means that the set of viable axes of rotation can be specified by some $\mathbf{v}\alpha$ representing the rotation of the unit vector of the axis of rotation $\mathbf{v}_0$ about a rotation axis $\hat{\mathbf{y}} - \hat{\mathbf{x}}$ by some angle α. The expression of axes of rotation as a function of α is given by

$$\mathbf{v}(\alpha) = \mathbf{R}(\alpha, \hat{\mathbf{y}} - \hat{\mathbf{x}})\mathbf{v_0} \tag{116}$$

Here $\mathbf{R}(\alpha)$ represents a rotation transformation of angle α about the vector $\hat{\mathbf{y}} - \hat{\mathbf{x}}$. Figure 20 illustrates the relationship of the angle of rotation $\psi(\alpha)$ as $\mathbf{v}(\alpha)$ is changed.

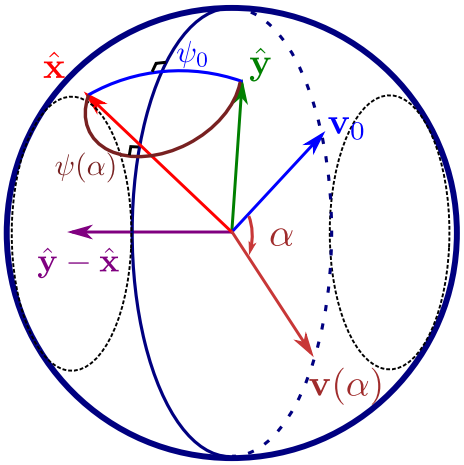


Figure 19. This demonstrates the different angles of rotation given two viable axes of rotation to rotate $\hat{\mathbf{x}}$ to $\hat{\mathbf{y}}$. $\mathbf{v}_0$ and ψ_0 is the geodesic rotation, $\mathbf{v}(\alpha)$ and $\psi(\alpha)$ represent another candidate.

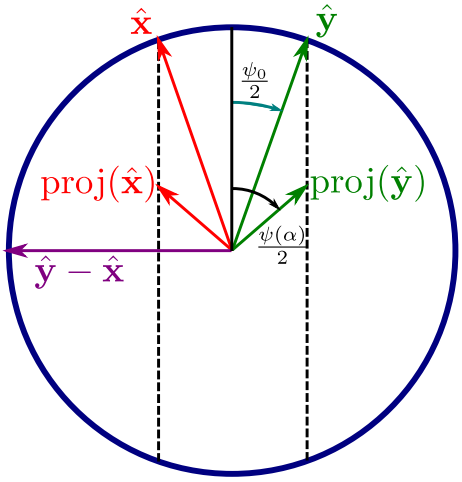


Figure 20. This representation depicts the projection of the vectors as the vector describing the axis of rotation is changed.

In this figure, the unit vectors are projected onto the plane that is normal to the axis of rotation vector $\mathbf{v}(\alpha)$.

For this analysis, the positive x axis is taken to mean in the direction of $\hat{\mathbf{x}} - \hat{\mathbf{y}}$ and the positive y axis is taken to mean in the direction of $\hat{\mathbf{x}} + \hat{\mathbf{y}}$. Based on the way the vectors are projected, $\hat{x}$ and $\hat{y}$ are reflections across the y axis.

The x component of $\hat{y}$ in this reference is given by $\sin(\frac{\psi_0}{2})$ and the y component is given by $\cos(\frac{\psi_0}{2})$. As the vector of rotation is changed, the value of the x component is constant, but the value of the y component changes to $\cos(\alpha)\cos(\frac{\psi_0}{2})$. This allows the calculation of the tangeant of the new rotation angle as

$$\tan\left(\frac{\psi}{2}\right) = \frac{\sin\left(\frac{\psi_0}{2}\right)}{\cos(\alpha)\cos\left(\frac{\psi_0}{2}\right)} = \frac{\tan\left(\frac{\psi_0}{2}\right)}{\cos(\alpha)}. \tag{117}$$

This yields the expression for the angle of rotation as a function of α.

$$\psi(\alpha) = 2\arctan\left(\frac{\tan\left(\frac{\psi_0}{2}\right)}{\cos(\alpha)}\right) \tag{118}$$

This equation does have a singularity in the calculation if $\cos(\alpha)$ is zero, but the arctan is defined for this value and gives a value of $\pm\frac{\pi}{2}$ resulting in a rotation $\psi(\alpha) = \frac{\pi}{2}) = \pm\pi$. This represents the rotation when the angle of rotation is coplanar with the vectors to rotate. Consequently, some care must be given for α close to $\pi 2$.

7.4.3 Two-Pair Estimation of Rotation

To be able to estimate the global rotation, two pairs of corresponding rotation points are taken from the test sensor and the reference sensor respectively. These measurements are denoted $\{\mathbf{x}_1, \mathbf{y}_1\}$ and $\{\mathbf{x}_2, \mathbf{y}_2\}$. These values are sampled from the available data being used by the optimization and are selected to be unique.

From these values, the geodesic rotation for each pair be identified as $\mathbf{v}_0^{(1)}$ and $\mathbf{v}_0^{(2)}$ with rotation angles $\psi_0^{(1)}$ and $\psi_0^{(2)}$. From this, values α_1 and α_2 are discovered

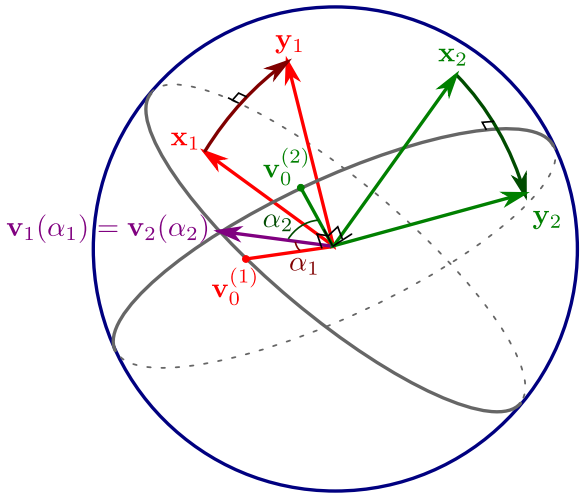


Figure 21. The depiction of the two-pair estimator is illustrated. Vector $\mathbf{x}_1$ rotates to $\mathbf{y}_1$ and $\mathbf{x}_2$ is rotated to $\mathbf{y}_2$. The two hemispheres represent the set of viable rotation axes to perform the two rotations. The unit vectors in the direction of the cross products are also illustrated as $\mathbf{v}_0^{(1)}$ and $\mathbf{v}_0^{(2)}$, and the angles of rotation α_1 and α_2 are depicted. Finally, the common vector of rotation is given by $\mathbf{v}_1(\alpha_1) = \mathbf{v}_2(\alpha_2)$.

such that $\mathbf{v}_1(\alpha_1) = \mathbf{v}_2(\alpha_2)$. An illustration of this is depicted in Figure 21.

To find this value, identify two new vectors $\mathbf{w}_1$ and $\mathbf{w}_2$.

$$\mathbf{w}_i = \frac{\hat{\mathbf{y}}_i + \hat{\mathbf{x}}_i}{\|\hat{\mathbf{y}}_i + \hat{\mathbf{x}}_i\|} \tag{119}$$

Since $\mathbf{v}_0$, $\hat{\mathbf{y}} - \hat{\mathbf{x}}$, and $\mathbf{w}$ are all mutually orthogonal vectors, so this allows an efficient expression for $\mathbf{v}(\alpha)$.

$$\mathbf{v}(\alpha) = \cos(\alpha)\mathbf{v}_0 + \sin(\alpha)\mathbf{w} \tag{120}$$

This provides three equations with two unknowns, so the system could be over-complete. Because the values of $\mathbf{v}(\alpha)$ describe hemispherical circles on a unit sphere, this constrains the underlying solution set of the problem. There are only two possibilities: the circles are identical and all values are valid solutions, or they intersect in exactly two points that are symmetrically located on the sphere. The function for $\psi(\alpha)$ provides rotations of opposite signs given the different vector solutions, again because of the double-covering nature of the solution space over SO(3). To ensure

that the circles are not identical, a test is executed to ensure that the difference vectors are sufficiently different by comparing their dot product. Given some small value ϵ, ensure that the following holds.

$$|\mathbf{w}_1 \cdot \mathbf{w}_2| \leq 1 - \epsilon \tag{121}$$

Another important consideration is to make sure that there is some rotation observed between the pair of vectors. For every rotation, there are two points that are fixed points along the axis of rotation. For points distributed on the surfaces of two spheres, points close to the axes of the rotation will be rotated by a very small amount. To distinguish this condition, the following test can be used.

$$|\hat{\mathbf{x}} \cdot \hat{\mathbf{y}}| < 1 - \epsilon \tag{122}$$

In this case, the epsilon can be chosen as the noise level increases or it can be set at a fixed value. It becomes increasingly difficult to identify fixed points as the noise level increases and in the selection there can be some bias for points with increased noise. This estimate, however, is taken to be heuristic and mainly serves to start the more complete optimization. This method is expected to be a poor estimator in systems with noisy measurements.

Another condition must also be considered when all pairs of vectors are identical and there is no rotation. This is important mostly because if the rotations are aligned, the system could get caught in an infinite loop looking for suitable vector pairs. To overcome this, the system could compare the conditions described and find the example within a specified number of sampled vectors that gives the lowest value and use those pairs of points. If a particular point meets the conditions, the program can stop looking. Otherwise, it checks some number of values and takes the best one that it sampled.

The system of equations that are used for calculation is expressed by the following along with a Jacobian calculation.

$$f_{\text{r-eqn}} = \cos(\alpha_1)\mathbf{v}_0^{(1)} + \sin(\alpha_1)\mathbf{w}_1 - \cos(\alpha_2)\mathbf{v}_0^{(2)} + \sin(\alpha_2)\mathbf{w}_2 \tag{123}$$

$$\frac{\partial f_{\text{r-eqn}}}{\partial \boldsymbol{\alpha}} = \begin{bmatrix} -\sin(\alpha_1)\mathbf{v}_0^{(1)\intercal} + \cos(\alpha_1)\mathbf{w}_1^{\intercal} \\ \sin(\alpha_2)\mathbf{v}_0^{(2)\intercal} - \sin(\alpha_2)\mathbf{w}_2^{\intercal} \end{bmatrix}^{\intercal} \tag{124}$$

The solutions for α_1, α_2 are found using the Python `fsolve` function. For programmatic reasons, the α vector is augmented with a zero value to match the dimensions of the optimization vector and the equation number.

An alternative technique was used that performed an optimization calculation using the L-2 norm. It was also successful. The equations used for this are below. For this, *zeta* is a symbol representing an equation to simplify the notation.

$$\zeta = \cos(\alpha_1)\mathbf{v}_0^{(1)} + \sin(\alpha_1)\mathbf{w}_1 - \cos(\alpha_2)\mathbf{v}_0^{(2)} + \sin(\alpha_2)\mathbf{w}_2 \tag{125}$$

$$f_{\text{r-est}} = \|\zeta\|^2 \tag{126}$$

$$\frac{\partial f_{\text{r-est}}}{\partial \boldsymbol{\alpha}} = 2\zeta^{\intercal}\frac{\partial \zeta}{\partial \alpha} \tag{127}$$

In the Jacobian equation, the partial derivative of ζ with respect to α is a 3×2 matrix where the columns represent the partial derivatives of ζ with respect to the angles α_1 and α_2 respectively. These values can be calculated with simple trigonometry derivatives and so the details are omitted.

After finding the appropriate α values, the vectors corresponding to the axis of rotation should be identical. The common rotation vector can be calculated using either one of them.

$$\hat{\mathbf{v}} = \cos(\alpha_1)\mathbf{v}_0^{(1)} + \sin(\alpha_1)\mathbf{w}_1 \tag{128}$$

$$= \cos(\alpha_2)\mathbf{v}_0^{(2)} + \sin(\alpha_2)\mathbf{w}_2 \tag{129}$$

Then the values of the rotation values are calculated using Equation 118 to give ψ_1 and ψ_2. Ideally, these values should be identical representing an identical amount of rotation, but in the presence of noise, these values are expected to be similar but not identical. To select an amount of rotation to use in particular, the average is taken.

$$\psi = \frac{\psi_1 + \psi_2}{2} \tag{130}$$

The performance of this method is dependent upon the amplitude and the nature of the noise. This is especially true as a random set of corresponding vectors are chosen and the values are assumed to be true. It is to be remembered, however, that this is only intended to be used as an initial estimate of a rotation that will be used as a parameter in a full rotation optimization function.

7.5 Rotation Metric

In order to compare the goodness of a rotation, a metric needs to be defined. In a paper by Du Q. Huynh, six different metrics were compared[33]. To understand the metric selection, a few mathematical properties are described.

The metric, μ, respects the topology of SO(3) if, given $\mathbf{R}_n, \mathbf{R} \in \mathrm{SO}(3)$,

$$\mu(\mathbf{R}_n, \mathbf{R}) \to 0 \iff \mathbf{R}_n \to \mathbf{R}. \tag{131}$$

A metric is said to be *bi-invariant* if, given $\mathbf{R}_1, \mathbf{R}_2, \mathbf{R}_3 \in \mathrm{SO}(3)$,

$$\mu(\mathbf{R}_1\mathbf{R}_2, \mathbf{R}_1\mathbf{R}_3) = \mu(\mathbf{R}_2, \mathbf{R}_3) \tag{132}$$

$$\mu(\mathbf{R}_2\mathbf{R}_1, \mathbf{R}_3\mathbf{R}_1) = \mu(\mathbf{R}_2, \mathbf{R}_3) \tag{133}$$

If only the first of these holds, the metric is said to be left-invariant; if only the second holds, it is right-invariant.

Two metrics, μ_1, μ_2 are *boundedly equivalent* if there exist $a, b \in \mathbb{R}^+$ such that, for all $\mathbf{R}_1$, $\mathbf{R}_2 \in \mathrm{SO}(3)$,

$$a\mu_1(\mathbf{R}_1, \mathbf{R}_2) \leq \mu_2(\mathbf{R}_1, \mathbf{R}_2) \leq b\mu_2(\mathbf{R}_1, \mathbf{R}_2). \tag{134}$$

Five of the six were bi-invariant metrics that respect the tolopolgy on SO(3), and four of those are mutually boundedly equivalent. This means that any of those four can be used somewhat interchangeably and give similar behavior. The one chosen for this analysis is given by

$$\arccos\left(|\mathbf{q}_1 \cdot \mathbf{q}_2|\right) \tag{135}$$

In this equation, $\cdot$ represesnts a vector dot-product of vectors representing the unit quaternions. The application of the absolute value of the dot product removes the problem of having two quaternion representations for the same rotation. This works by symmetry. The angle to be returned by the arccos is limited to be in the range $[0, \pi/2]$, which means that the rotation distance is bounded.

First, it should be recognized that this equation is defined for all rotation vectors $\mathbf{q}_1, \mathbf{q}_2 \in \mathrm{SO}(3)$. This is true since the dot product of the quaternion unit vectors must have a value $-1 \leq \mathbf{q}_1 \cdot \mathbf{q}_2 \leq 1$. With the absolute value, this means that the values of the argument must be in $[0, 1]$. These values are all within the domain of arccos(). It can be shown that this is a proper metric by testing the axioms.

- $d(x, y) > 0$: This is trivial due to the range of the arccos() function.

- $d(x, y) = 0 \iff x = y$: It could be claimed this is violated because $d(x, -x) = 0$ for all x, but the metric is not defined on the space of quaternions but on the space SO(3), over which the quaternions are a double covering, so this is also true. In other words, it respects the topology.

- $d(x, y) = d(y, x)$: This is true because the dot product gives $x \cdot y = y \cdot x$ when x, y are real-valued vectors.

- $d(x, z) \leq d(x, y) + d(y, z)$: For this, recognize that $|x \cdot y|$ is $x \cdot y$ when the value is positive, and $x \cdot -y$ when the value is negative. With the quaternion representation, these two vectors represent the same rotation. This means that these terms represent $\cos(\theta_{x,y})$, or the cosine of the smallest angle between the two vectors. Applying the arccos function to each term, this gives $\theta_{x,z} \leq \theta_{x,y} + \theta_{y,z}$. This is true when the angles represent the shortest angles between the corresponding vectors, therefore the triangle inequality holds.

It is also beneficial to, given a specific rotation vector, what is the expected value of the rotation difference given another random vector. To find this, assume without loss of generality that a particular rotation $\mathbf{R}_1$ is expressed by the vector $\mathbf{v}_1 = (\alpha, 0, 0)$. This represents the vector portion of a quaternion, so it is a rotation of $2 \arcsin(\alpha)$ about the axis $(1, 0, 0)$ and is expressed by the quaternion

$$\mathbf{q}_1 = \left(\sqrt{1 - \alpha^2}, \alpha, 0, 0\right). \tag{136}$$

To find the expectation, an average of the metric is found against all other vector components of other quaternions. This means that given a vector $\mathbf{v}_2 = (r, \theta, \phi)$. Given values of (r, θ, ϕ), the quaternions can be given by

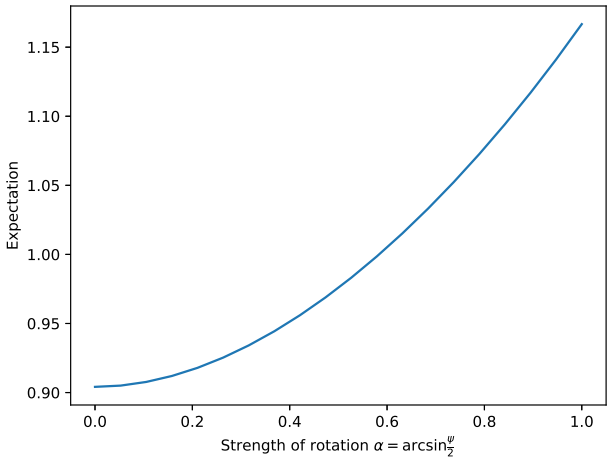


Figure 22. Depiction of the expectation of metric measurements given a particular strength of reference rotation parameterized by α

$$\mathbf{q}_2 = \left(\sqrt{1 - x^2 - y^2 - z^2}, x, y, z\right) \tag{137}$$

$$x = r\sin(\theta)\cos(\phi) \tag{137a}$$

$$y = r\sin(\theta)\sin(\phi) \tag{137b}$$

$$z = r\cos(\theta). \tag{137c}$$

$$\mathbb{E}\left[\mu(\mathbf{R}_1, \mathbf{R}_2)\right] = \frac{3}{4\pi} \iiint\limits_{(r,\theta,\phi)\in B(1)} \arccos\left(|\mathbf{q}_1 \cdot \mathbf{q}_2|\right) r^2 \sin(\theta) \mathrm{d}r\mathrm{d}\theta\mathrm{d}\phi \tag{138}$$

Here $B(1)$ represents the unit ball, i.e. all the points within the closed unit sphere. Results of a numeric calculation of these values is given in Figure 22.

The variance is calculated by the following.

$$\mathrm{Var}\left[\mu(\mathbf{R}_1, \mathbf{R}_2)\right] = \frac{3}{4\pi} \iiint\limits_{(r,\theta,\phi)\in B(1)} \left(\arccos\left(|\mathbf{q}_1 \cdot \mathbf{q}_2|\right) - \mathbf{E}\mu_\alpha\right)^2 r^2 \sin(\theta) \mathrm{d}r\mathrm{d}\theta\mathrm{d}\phi \tag{139}$$

The variance is depicted in Figure 23.

From these two figures, it can be seen that stronger rotations have an increased expectation for the metric. It can also be seen that the variance also increases quite

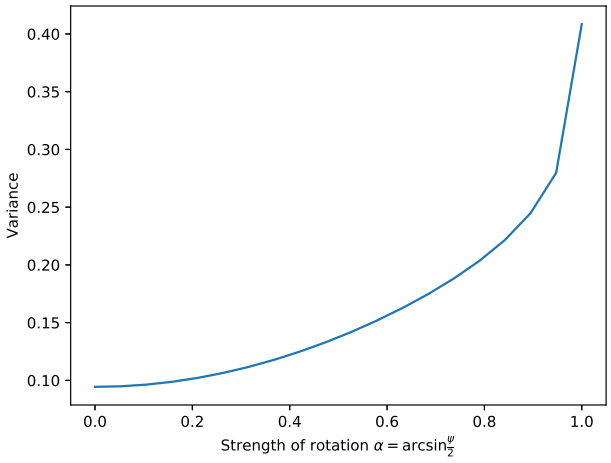


Figure 23. Depiction of the variance of metric measurements given a particular strength of reference rotation parameterized by α

a bit as the rotations become stronger. This also is helpful to show that, while the largest error value possible is $\pi/2$, the average distance of a randomly chosen vector has a distance of about 1.

7.6 Evaluation

Each of the techniques described above is used for testing. The affine method is a little bit different since it does not generate a rotation itself, but rather generates a matrix and offset. Specific elements related to comparing this method to the other methods are also described.

To separate the assessment of this section from the earlier step in the previous chapter, new simulated data is generated. First, two rotations are randomly generated by selecting a random angle of rotation $\psi \in [-\pi, \pi]$, and a random orientation vector on the surface of $\mathbb{R}^3$ which values are then converted to θ, ϕ.

Random path values are created by using the same method described in the last chapter. Each of the paths is rotated by the random rotation and independent random noise is generated and added to the signals for each sensor. In these experiments,

the noise standard deviation is always assumed to be identical for both sensor measurements. From this, tests are performed using each of the outlined methods over each of the different path types.

The same method for estimating initial conditions follows the method outlined in Section 7.4.3. This is also used for the initial estimate for the matrix in the affine section. In that algorithm, the initial estimate for the center is the zero vector.

The number of points used were specified by $\{10, 30, 100, 300, 1\,000, 3\,000, 10\,000\}$. Noise was added with a standard deviation of $\{0, 0.00316, 0.01, 0.0316, 0.1, 0.316, 1\}$. Since the noise is applied once to each of the sensor measurements, the noise is the sum of the two noise applications. This means that the signal-to-noise ratios are $\{47\text{dB}, 37\text{dB}, 27\text{dB}, 17\text{dB}, 7\text{dB}, -3\text{dB}\}$ respectively.

7.6.1 Execution Time

The execution time of all methods is of approximately the same computational complexity. The affine method is a little less efficient for low numbers of samples. This could be because the lack of information can lead it to drift more, since it has a larger number of degrees of freedom. The resulting execution time when conditioned on simulation type are depicted in Figure 24

From this it can be seen that the L^2 method (L2) gives the fasted calculation. The affine method (AFF) is similar, but slightly slower. The projected-L^2 (PL2) and the central-angle method (CAD) are slower still. The CAD2 method takes about four times longer than the L2 method for 10^4 data points.

Even then, the slowest methods are still able to calculate all values in less than 10s for 10^4 sample points, which is still well within tolerable limits for finding a common sensor alignment in an initialization step.

The execution time was indistinguishable as a function of the path type. Results are shown in Figure 25. There is only some slight spread for very low numbers of

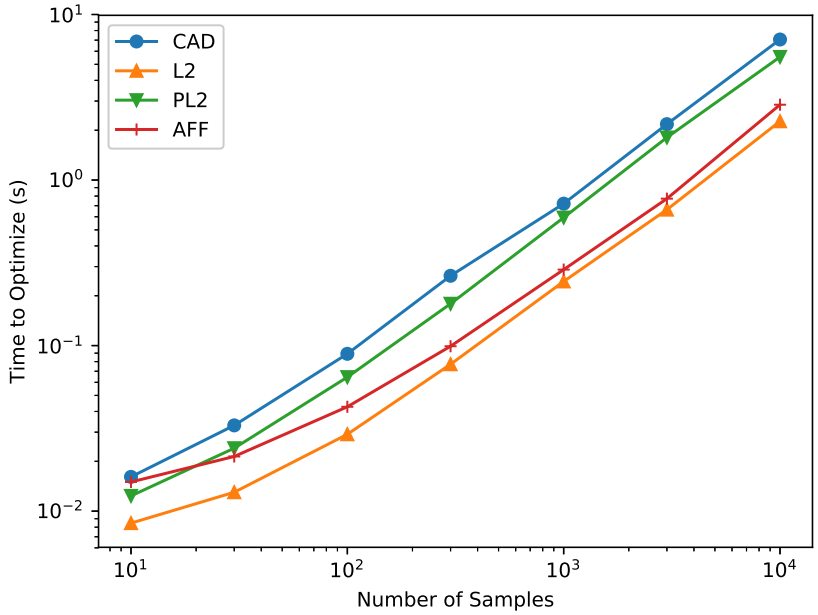


Figure 24. The runtime for each of the described methods is shown. CAD is the central-angle distance, L2 is the L^2-based optimization, PL2 is the projected L^2 optimization, and AFF is the affine transform method.

data points. In this region, random walk is a poor depiction and random points is best as it is more likely to give a diverse selection of points.

7.6.2 Projection of a Matrix to a Rotation Matrix

Because the affine method provides a matrix that is not necessarily a rotation matrix, it is necessary to generate a rotation matrix to use in the comparison to be able to compare it with the other techniques. To do this, a projection of the matrix to the space of SO(3) can be found by the optimization.

$$\mathbf{R} = \underset{\mathbf{R} \in SO(3)}{\arg\min} \|\mathbf{M} - \mathbf{R}\|_F^2 \tag{140}$$

Here $\|\cdot\|_F$ represents the Frobenius norm. A closely related problem finds a matrix in the real-valued orthogonal matrix group. This case also includes reflections, i.e. $\det(A) = \{-1, 1\}$. The solution to this problem is known [31] and has a closed-form

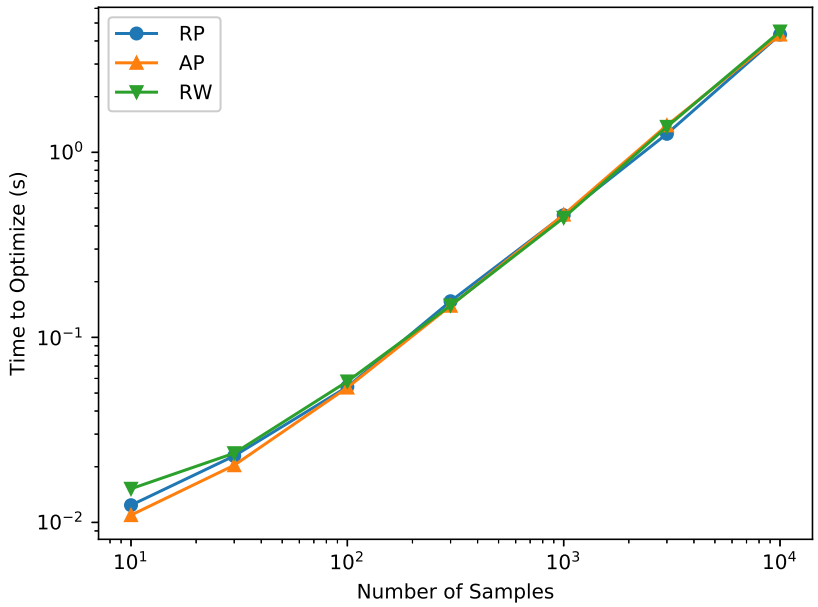


Figure 25. The runtime by path type is shown. RP is random-points, AP is axial path, and RW is the random walk path.

expression.

$$\mathbf{R} = \mathbf{M}\left(\mathbf{M}^{\intercal}\mathbf{M}\right)^{-1/2} \tag{141}$$

For the purposes of notation, $P(\mathbf{A})$ is taken to represent the projection of the matrix $\mathbf{A}$ to a matrix in the set of orthogonal matrices. Due to the nature of the problem, it is expected that this matrix is orientation-preserving, so it should also be a member of $\mathbf{SO}(3)$.

From a specific affine transformation estimation $\mathbf{A}$ and $\mathbf{o}$, the nearest rotation can be calculated and used in a comparison with the other rotation methods. Because the data was not shifted between the two versions, the offset is expected to be 0. With this method, two additional metrics are provided: $\|\mathbf{A} - P\mathbf{A}\|_F$ and $\|\mathbf{o}\|$.

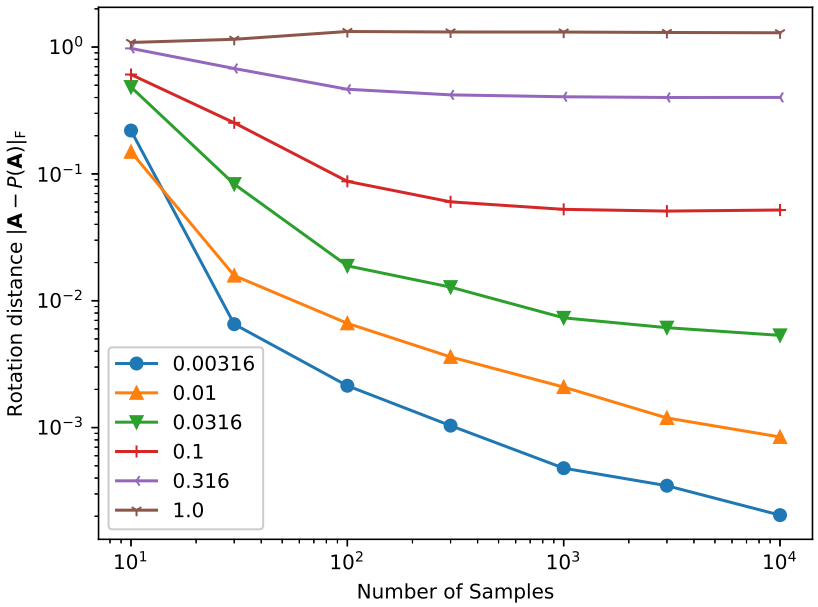


Figure 26. The distance between the matrix **A** and its projection on SO(3) as measured by the Frobenius norm. This is a representation of how much the resutig matrix resembles a rotation matrix.

7.6.3 Residuals of Affine Method

Because the affine method uses a projection to find the nearest rotation, it is useful to assess how much residual is realized by this projection.

7.6.3.1 Projection Residual Between Matrix Estimate and Its Projection

To show the behavior of the residual between the affine estimate and the projection of the rotation, the projection is calculated and the Frobenius norm is calculated. The values are shown as a function of the number of samples and the amount of noise. Results from the experiments are in Figure 26.

It is difficult to attach these measures to an intuitive understanding of how similar the resulting matrices are to a rotation based on the values from the Frobenius error, but this does show that with lower noise and with higher numbers of samples the resulting matrices are increasingly similar to rotation matrices. It is useful to

remember that this technique is applied, in part, for allowing further adjustment to help the measurements from a test sensor to more closely match the reference sensor. The benefits of this technique are unlikely to be realized in a system that does not test the interaction between the two portions.

It can be seen, however, that for noise values of 1 the rotation matrices don't seem to become more similar to rotation matrices, and only a slight trend towards rotation matrices appears for noise of 0.316. Because the signal strength itself is 1, this is not an unexpected result.

7.6.3.2 Offset Value From Affine Method

This section evaluates the offsets. The true offset should be zero, since there is no shift applied in the process rotating from one sensor measurement to the other. In practical use, this offset should be seen as a corrective measure to account for the differences in the offsets between a test sensor and the reference sensor. In this case, there is no such offset applied, so this result could be seen as how much offset error is expected as a function of noise and the number of samples being used. The results are in Figure 27.

It is quite surprising just how much offset error is observed for small numbers of samples and large amounts of noise. With an SNR of 7dB and at 100 samples, the offset is within 1% of the signal. The same is true for an SNR of 2dB for 300 samples.

7.6.4 Rotation Accuracy

To measure the accuracy of the rotation, the composite of the two rotations is compared to the rotation estimated by the technique. The measure is taken using the calculation described in Section 7.5. In these measurements, the values presented are based on the random-walk path.

The first thing to look at is the accuracy of the initial estimate method. These results are in Figure 28.

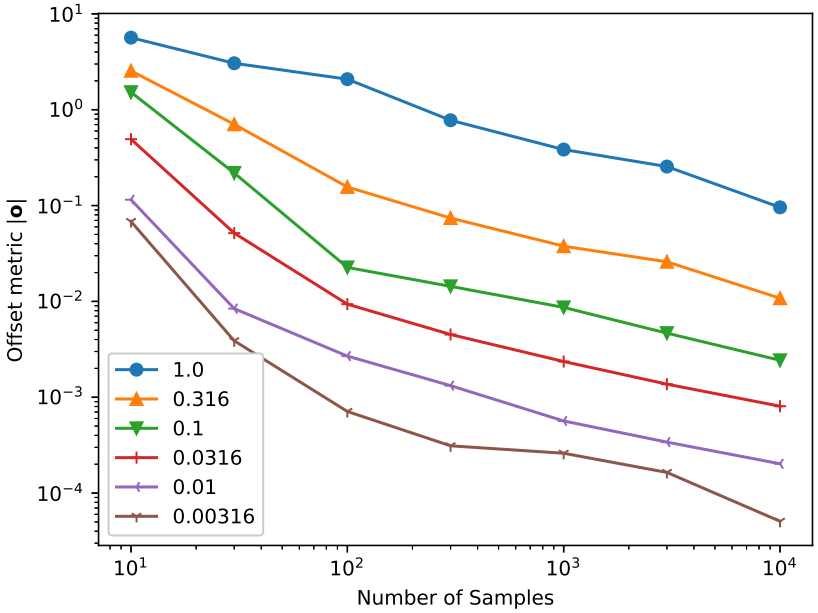


Figure 27. The amplitude of the offset when estimating the shift by an affine transformation. No shift was applied to the simulation data so the true value should be zero.

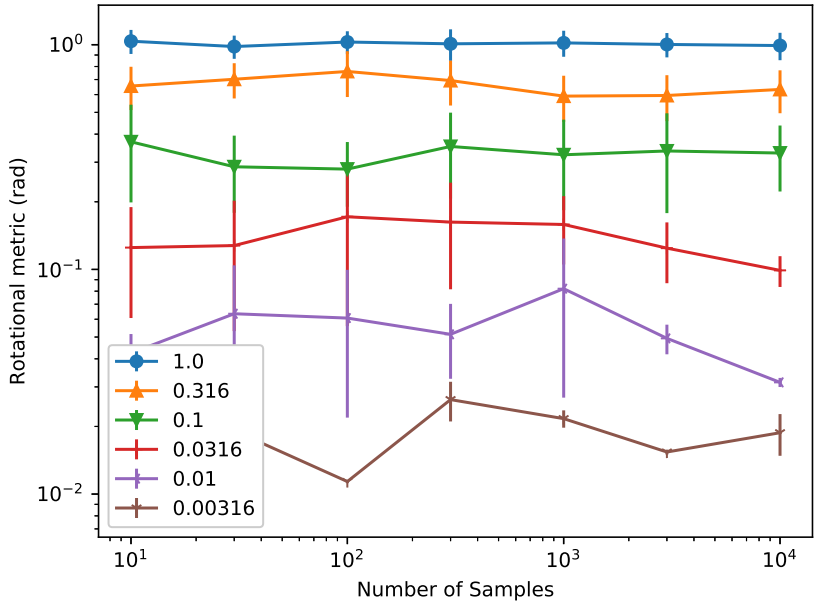


Figure 28: Rotation accuracy as given by the two-vector estimate technique.

Table 2. Table of statistics of the mean and standard deviation of estimates from calculating two-vector estimates under various noise conditions.

Noise σ	Error μ	Error σ
1.0	1.01005914e+00	1.29820622e-01
0.316	6.61143568e-01	1.45309408e-01
0.1	3.25448049e-01	1.30135622e-01
0.0316	1.38361670e-01	6.25934870e-02
0.01	5.44079745e-02	2.43293578e-02
0.00316	1.73053076e-02	2.00943941e-03
0.0	6.15480923e-09	2.82670184e-16

This plot shows the mean and variance estimates. All the curves were generated by using points from the random points path since the path is essentially irrelevant in this case. Only weak conditions were placed on the fitness of selection (such as ensuring that the two rotations did not define the same set of rotation axes.) The curves are essentially flat since only two points are used for the estimation regardless of the amount of available data. The plot is mainly included to help visualize the expectation. For an SNR of -3dB, the estimates are no better than random guessing; as long as the signal is stronger than the noise, the estimate should be an improvement.

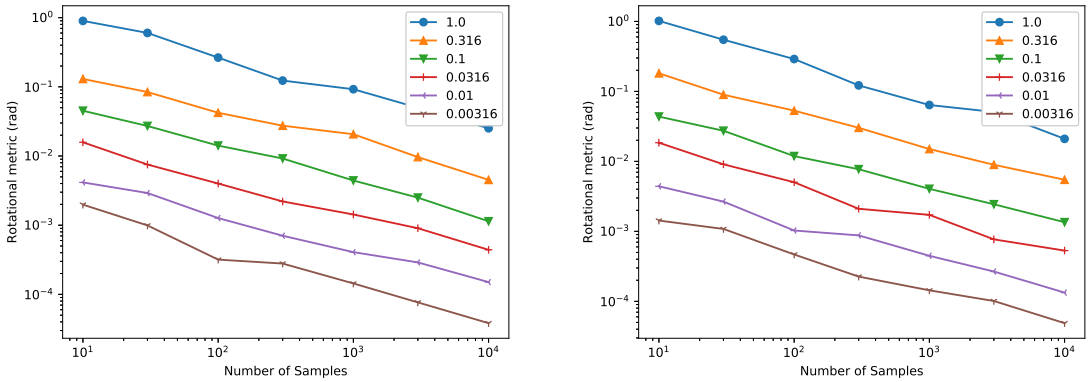
The average time required to perform an initial estimate is 2.8535ms and is independent of the noise or the sample size.

The statistics are summarized in Table 2. In the table, the noiseless case is included as an illustration that the algorithm is accurate in the absence of noise.

The performance of the three rotation-based methods are all roughly equivalent. The two L^2 methods are essentially indistinguishable in terms of accuracy. This can be seen more easily in the comparison of the different methods given a specific noise estimate as depicted in Figure 30.

The rotation estimate using the affine transformation seems to have a limit at about 0.03. This demonstrates that in direct rotation tasks, the other methods are preferable to giving an accurate rotation.

For a noise of 1.0 and with 10 samples, the rotation metric provides a value that



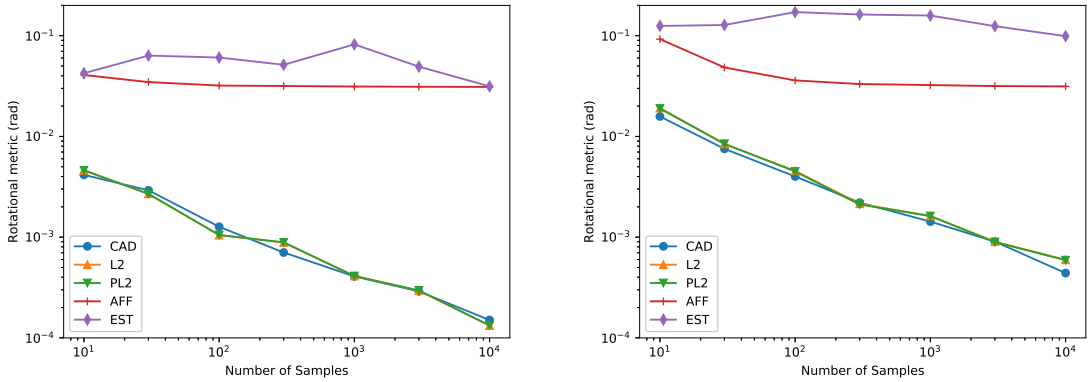
(a) Central-angle distance method

(b) L^2 method

(c) Projected L^2 method

(d) Affine method

Figure 29. Measurements of the distance between the rotations specified by the optimization and the true rotation between the points. In the case of the affine method, the measurement was taken with respect to the projection of the matrix estimate to the nearest rotation matrix.



(a) Comparison with noise $\sigma = 0.01$ (b) Comparison with noise $\sigma = 0.0316$

Figure 30. Comparison of the rotation metric across the three rotation-based techniques

is no better than a random rotation estimate; all the rest of the rotations are at least somewhat informative. This is true for all methods. This is not unexpected since the noise is strong enough to completely mask the signal.

When considering the rotational accuracy based on the path type, it can be seen that the path seems to have little or no effect in the accuracy of the rotations. The nature of the rotation system means that the rotation methods are largely path invariant. The results are in Figure 31.

7.7 EMF Removal

Now that the sensors have a common orientation, it is possible to remove the component created by Earth's magnetic field. Any magnets in the area will change the accuracy of this removal as it adjusts the field experienced on each of the sensors. In fact, the magnet to be tracked has a small effect on the fifth sensor, but this measurement is much weaker. The expression can be calculated as follows. Given two sensor positions with radius distance from the magnet of ρ_1 and ρ_2, the ratio of the B-fields are constrained by the following values.

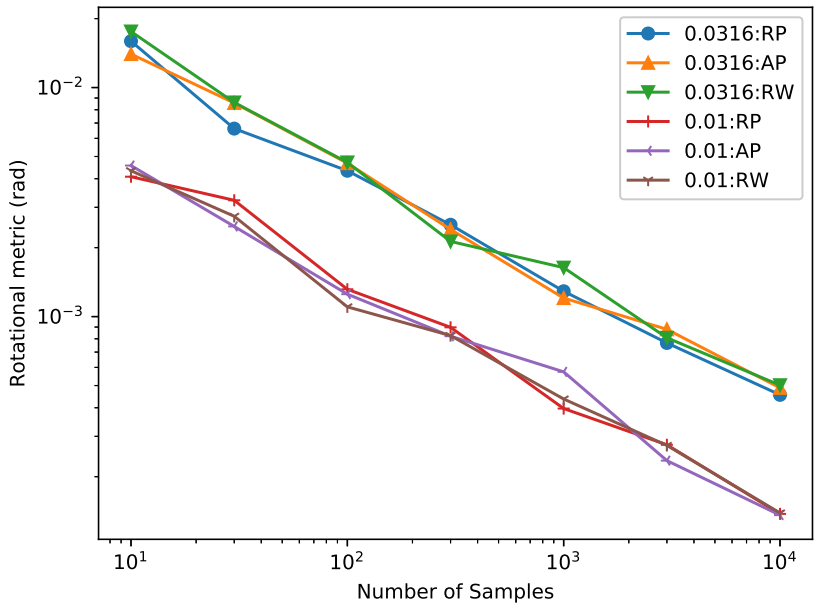


Figure 31. Comparison of the rotation accuracy as a function of the number of samples for noise values of $\sigma = 0.01$ and $\sigma = 0.0316$. The path types are represented as RP: random points, AP: axial path, RW: random walk

$$\frac{1}{2}\frac{1}{(\rho_2/\rho_1)^3} \leq \frac{\beta_2}{\beta_1} \leq 2\frac{1}{(\rho_2/\rho_1)^3} \tag{142}$$

The largest value occurs when the magnetic moment is aligned in the direction of sensor 2 and the magnetic moment is orthogonal to the vector in the direction of sensor 1. The smallest value occurs when the magnetic moment is orthogonal to the direction of sensor 2 and is aligned with sensor 1. This means if the fifth sensor is three times further away than the sensor magnet than a measurement sensor, the effect of the magnet on the fifth sensor is between $1/54$ and $2/27$ of the effect on the measurement sensor.

Any additional magnetic fields that are approximately identical between the different sensors can also be removed with this method. Errors only occur in the presence of a magnetic field that has different directions or strengths between the sensors.

Once the rotation value has been identified, the measurements at each of the

sensors is provided by the following equation and will be processed accordingly.

$$\mathbf{b}_i = \mathbf{R}_i \hat{\mathbf{A}}_i (\mathbf{b}_{\text{meas}}^{(i)} - \hat{\mathbf{o}}_i) - \hat{\mathbf{A}}_{\text{ref}} (\mathbf{b}_{\text{meas}}^{(\text{ref})} - \hat{\mathbf{o}}_{\text{ref}}) \tag{143}$$

All the values of $\mathbf{b}$ are now expected to represent measurements that are aligned according to some common orientation and that are calibrated so that the measurements represent orthogonal measurements that are properly calibrated for gain and bias.

CHAPTER VIII

SENSOR POSITION ESTIMATION

8.1 Aim

The target of this section is to attempt to find the position of sensors given a set of calibrated and oriented magnetic field measurements. It is expected that prior calibration steps are complete, that the sensors are in a common orientation, and that the fifth sensor is used to remove earth's magnetic field and other interfering fields from the tracking sensors.

Training measurements for tracking sensor position are taken in the presence of a magnet with an known magnetic moment and that the calibrated sensor measurements have a proper alignment with the implied orientation space indicated by the magnet parameters.

Initially, an effort was made to perform a single optimization to find the orientation rotation and sensor position values simultaneously, but the corresponding optimization system failed to converge. To attempt to find a way to find both orientation and sensor unknowns, the problem was split into two distinct parts with the intention of combining them. This chapter covers solving for the sensor positions under a known orientation. The next chapter covers orientation given known sensor position. The following chapter then outlines a method of solving the system for both unknowns.

8.2 Magnetic Dipole Equation

To be able to identify the position of the sensors (or similarly finding the position of the magnet given known sensor positions) is related by using the magnetic dipole equation.

Assuming that a magnetic sensor is far enough away from the magnet, the magnetic field generated by a magnet with a magnetic moment $\mathbf{m}$ at a position $\mathbf{a}$ on a sensor at position $\mathbf{s}$ is given by the following equation:

$$\mathbf{b}(\mathbf{s}, \mathbf{a}, \mathbf{m}) = \frac{\mu_0}{4\pi} \frac{3\left((\mathbf{s}-\mathbf{a})\cdot\mathbf{m}\right)(\mathbf{s}-\mathbf{a}) - \|\mathbf{s}-\mathbf{a}\|^2 \mathbf{m}}{\|\mathbf{s}-\mathbf{a}\|^5}. \tag{144}$$

Since the magnets being tracked in this project are quite small and because the sensors are outside of the user's mouth, the far-field assumption is appropriate.

The magnetic moment $\mathbf{m}$ is a vector relationship describing the aligning torque on a magnet from an externally applied magnetic field. It is measured in $\mathrm{A} \cdot m^2$ or equivalently $\frac{\mathrm{N \cdot m}}{\mathrm{T}}$. The magnetic moment points in the direction of the north pole of the magnet.

The parameters of a magnet are specified by the residual induction ($\mathrm{Br_{max}}$) or the residual flux density. This is a parameter that can be taken from the hysteresis loop and it is measured in units of gauss or tesla. The relationship between this value and the magnetic moment is given by

$$\mathbf{m} = \mathrm{B_r} \frac{V}{\mu_0}. \tag{145}$$

Here, $\mathrm{B_r}$ is the residual flux density, V represents the volume of the magnet in m^3, and μ_0 is the permeability of free space.

$$\mu_0 = 4\pi \times 10^{-7} \frac{\mathrm{N}}{\mathrm{A}^2} \tag{146}$$

The magnets used in this project were neodymium disk magnets from K&J Magnetics, Inc. [37]. The dimensions of the magnet are available along with the relevant material properties and parameters as well as documentation of the orientation of the magnetic poles in relation to the magnet, providing the direction of the magnetic moment. This means it is straightforward to identify a magnetic dipole moment from the datasheet of a particular magnet.

For the purposes of this analysis, it is useful to express the magnetic field in spherical vector coordinates rather than rectangular vector coordinates. That is to say that the vector values are expressed in terms of unit vectors $\hat{\rho}, \hat{\theta}, \hat{\phi}$. It is important to recognize that the direction of these vectors is a function of the point at which the representation is being taken. Because of the changing frame of reference for the vector fields, it is important to make sure that the notation is clear.

In this work, a location as represented in spherical coordinates will be denoted as $\boldsymbol{\rho} = (\rho, \theta, \phi)$. The vector field (in this case an arbitrary vector field denoted $\mathbf{v}$) measured a location described using the spherical unit vectors $\hat{\rho}, \hat{\theta}, \hat{\phi}$ is given by

$$\mathbf{v}(\boldsymbol{\rho}) = (\varrho(\boldsymbol{\rho}), \vartheta(\boldsymbol{\rho}), \varphi(\boldsymbol{\rho})) = \varrho\hat{\rho} + \vartheta\hat{\theta} + \varphi\hat{\phi}. \tag{147}$$

For this equation, it is assumed that the magnet is located at the origin with the magnetic moment vector oriented in the positive z direction. If the magnet is in some other orientation, the frame of reference must be changed accordingly.

$$\mathbf{b}(\boldsymbol{\rho}) = \frac{\mu_0}{4\pi}\frac{\|\mathbf{m}\|}{\rho^3}\left(2\cos\theta\hat{\rho} + \sin\theta\hat{\theta}\right) \tag{148}$$

Here $\mathbf{r}$ represents the vector pointing from the magnet to the sensor and $r = \|\mathbf{r}\|$ represents the distance between the magnet and the sensor. It is useful to recognize that the term with $\hat{\phi}$ is zero, so all magnetic fields point radially outward. It can also be a little bit complicated to distinguish between spherical coordinates and spherical vectors with the unit vectors. To help distinguish these, a spherical vector will be denoted as $\boldsymbol{\rho} = (\rho, \theta, \phi)$.

8.2.1 Symmetry Conditions

To properly analyze the system, it is important to consider the symmetry conditions of the magnetic dipole equation. The first thing to recognize is that $\mathbf{b}$ is actually a function with respect to $\mathbf{r} = \mathbf{s} - \mathbf{a}$ and $\mathbf{m}$ and can be expressed more succinctly as

$$\mathbf{b}(\mathbf{r},\mathbf{m}) = \frac{\mu_0}{4\pi}\left[\frac{3(\mathbf{m}\cdot\mathbf{r})\mathbf{r}}{r^5} - \frac{\mathbf{m}}{r^3}\right]. \tag{149}$$

There is a reflective symmetry present from this equation. This can be demonstrated by setting $\mathbf{r} = -\mathbf{r}$.

$$\mathbf{b}(-\mathbf{r},\mathbf{m}) = \frac{\mu_0}{4\pi}\left[\frac{3\left(\mathbf{m}\cdot-\mathbf{r}\right)\left(-\mathbf{r}\right)}{r^3} - \frac{\mathbf{m}}{r^3}\right] \tag{150}$$

$$= \frac{\mu_0}{4\pi}\left[\frac{3\left(\mathbf{m}\cdot\mathbf{r}\right)\mathbf{r}}{r^5} - \frac{\mathbf{m}}{r^3}\right] = \mathbf{b}(\mathbf{r},\mathbf{m}) \tag{151}$$

Because of this, optimization of sensor positions based upon magnetic field measurements is not unique. Assuming that there is some prior information about the region of expected values, this can be accommodated and it is quite simple to find the alternate solution.

There is an additional and more troublesome symmetry to consider when $\mathbf{r} \perp \mathbf{m}$.

$$\mathbf{b}(\mathbf{r},\mathbf{m}|\mathbf{r}\perp\mathbf{m}) = \frac{\mu_0}{4\pi}\left[\frac{3(0)\mathbf{r}}{r^5} - \frac{\mathbf{m}}{r^3}\right] = -\frac{\mu_0}{4\pi}\frac{\mathbf{m}}{r^3} \tag{152}$$

This means that there is a circular symmetry around the hemisphere about the magnetic dipole. This symmetry is unstable because small perturbations can yield completely different estimates for $\mathbf{r}$ and thus estimates for $\mathbf{s}$. If $\mathbf{r}\cdot\mathbf{m} \approx 0$, the position estimate may be undependable.

Because this optimization is based upon estimating the parameter $\mathbf{r}$, it is better to be able to identify a potential problem from $\mathbf{b}$, but because of the behavior of $\mathbf{b}$ in this case, the test condition would be

$$\frac{\mathbf{m}^\intercal\mathbf{b}}{\|\mathbf{m}\|\|\mathbf{b}\|} = -1 + \epsilon \text{ for } \epsilon > 0 \text{ small.} \tag{153}$$

This represents a comparison of the cosine between the two vectors. The value of ϵ can be specified to identify a region in which the values are deemed to be unstable.

There is a similar symmetry along the axis of the dipole in the direction of $\mathbf{m}$, however this is not a concern as small deviations will yield small deviations in the estimates of $\mathbf{s}$. Having many samples can help to alleviate this, but it indicates that trying to perform estimation when the $\mathbf{r}$ vector is orthogonal to $\mathbf{m}$ is less likely to give meaningful estimates of directionality.

8.3 Direct Optimization Method

For the optimization, assume that all the sensors have been calibrated and are in common alignment using the orientation algorithm outlined in Chapter 7. To recognize the presence of a rotation, let $\boldsymbol{\Xi}$ represent a matrix in SO(3), the special orthogonal group, or the rotation group. It represents some global rotation that realigns the rotation of the sensors (which are all in common) to the reference orientation implied by the orientation of the magnet.

Let $\mathbf{b}_{\text{est}}(\hat{\mathbf{s}}_i, \mathbf{a}, \mathbf{m})$ represent the Cartesian estimate of the magnetic field measured at the estimated sensor position for the i-th sensor. Let $\mathbf{b}_i[k]$ represent measurements of the B-field taken at the i-th sensor after being processed by the sensor optimization and the orientation functions.

For the purposes of this optimization, it is assumed that $\mathbf{m}$ and $\mathbf{a}$ are constant argument parameters and the optimization is over $\hat{\mathbf{s}}$. Here, $\mathbf{r}_i = (\hat{\mathbf{s}}_i - \mathbf{a})$.

A viable objective function candidate can be given by

$$f_{\text{direct}}(\hat{\mathbf{s}}_i) = \sum_k \left\| \mathbf{b}_{\text{est}}(\hat{\mathbf{s}}_i, \mathbf{a}, \mathbf{m}) - \boldsymbol{\Xi}\mathbf{b}_i[k] \right\|^2. \tag{154}$$

The Jacobian is calculated using the following equations:

$$\frac{\partial f_{\text{direct}}(\hat{\mathbf{s}}_i)}{\partial \hat{\mathbf{s}}_i} = 2\left(\sum_k \left(\mathbf{b}_{\text{est}}(\hat{\mathbf{s}}_i, \mathbf{a}, \mathbf{m}) - \boldsymbol{\Xi}\mathbf{b}_i[k] \right) \right)^{\mathsf{T}} \frac{\partial \mathbf{b}_{\text{est}}}{\partial \hat{\mathbf{s}}_i} \tag{155}$$

$$\frac{\partial \mathbf{b}_{\text{est}}(\hat{\mathbf{s}}_i, \mathbf{a}, \mathbf{m})}{\partial \hat{\mathbf{s}}_i} = \frac{3\mu_0}{4\pi \|\mathbf{r}_i\|^5} \left[\frac{-5\mathbf{r}_i \mathbf{m}^{\mathsf{T}} \mathbf{r}_i \mathbf{r}_i^{\mathsf{T}}}{\|\mathbf{r}_i\|^2} + \mathbf{m}^{\mathsf{T}} \mathbf{r}_i \mathbb{I} + \mathbf{r}_i \mathbf{m}^{\mathsf{T}} + \mathbf{m}\mathbf{r}_i^{\mathsf{T}} \right] \tag{156}$$

It is practical to divide the results by the number of measurements. This equation is not as efficient as it could be since it is necessary to perform a rotation on each sample term. It is more efficient to be able to reduce the number of matrix multiplications performed. Because the value $\boldsymbol{\Xi}$ is a rotation matrix, $\boldsymbol{\Xi}^{-1} = \boldsymbol{\Xi}^{\mathsf{T}}$. Furthermore, the rotation could be moved to the $\mathbf{b}_{\text{est}}$ term to allow more efficient calculation.

$$f_{\text{direct}}(\hat{\mathbf{s}}_i) = \frac{1}{K} \sum_k \left\| \boldsymbol{\Xi}^{\mathsf{T}} \mathbf{b}_{\text{est}}(\hat{\mathbf{s}}_i, \mathbf{a}, \mathbf{m}) - \mathbf{b}_i[k] \right\|^2 \tag{157}$$

This allows the inverse rotation to be applied to the single estimate reducing the number of rotations from K to 1.

This same method can be applied to the Jacobian calculation as well.

$$\frac{\partial f_{\text{direct}}(\hat{\mathbf{s}}_i)}{\partial \hat{\mathbf{s}}_i} = 2 \left(\mathbf{b}_{\text{est}} - \overline{\mathbf{b}_i} \boldsymbol{\Xi}^{\mathsf{T}} \right) \frac{\partial \mathbf{b}_{\text{est}}}{\partial \hat{\mathbf{s}}_i} \tag{158}$$

In this equation, $\overline{\mathbf{b}_i}$ represents the average of the individual measurements. This significantly reduces the number of multiplications required. The only element that must be iterated over all the data is the averaging operation.

When the result from this calculation is returned, it may be necessary to adjust to the other solution. From the symmetry conditions outlined in Section 8.2.1, there is typically a single point of symmetry that exists as a reflection across the magnet position. To find this point, the new estimate of $\hat{\mathbf{s}}$ is found by flipping the signs of the terms and then subtracting twice the value of the magnet offset about the origin $\mathbf{a}$. The reason that twice the shift must be added is that the original estimate included the value of the offset. It is negated along with the reflection and so double the offset is applied to compensate.

$$\hat{\mathbf{s}}_{i,\text{alt}} = -\hat{\mathbf{s}}_i + 2\mathbf{a} \tag{159}$$

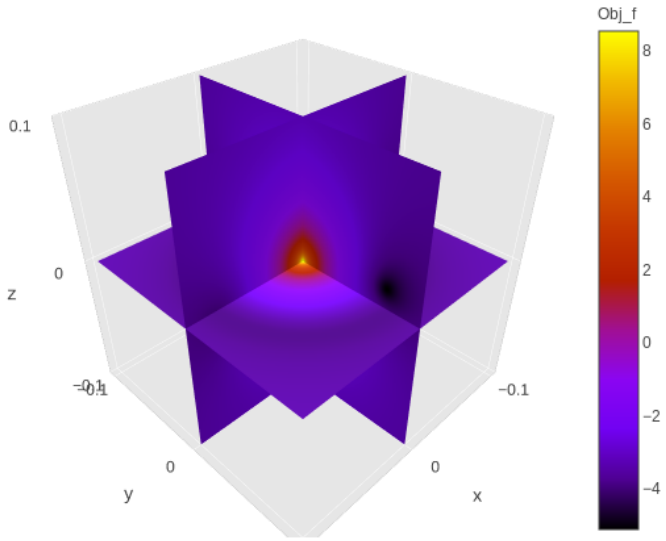
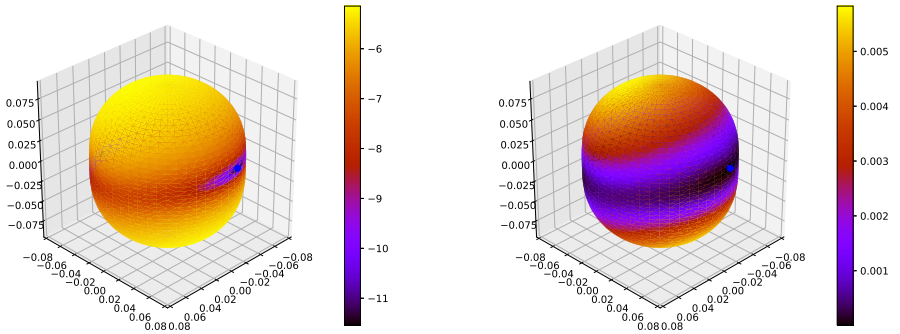


Figure 32. Depiction of the logartihm of the objective function. The true value is at the point $\mathbf{s} = [-0.02, 0.07, 0.015]$. The logarithm is used because of the singularity at the origin.

8.4 Initial Value Estimation

In order to properly be able to converge the objective function, it is important to provide a good initial condition. The reason that a good initial condition is so important can be seen in a representation of the solution space. This can be seen in Figure 32. Because the magnetic dipole equation has a singularity at the magnet location, the general gradient is outward. The fact that the objective function is not convex is clear from the depiction of the space. It is quite likely that, with a poor initial guess, the optimization will find a sensor position increasingly far away from the magnet.

To be able to counter this, a method based on seeking a minimal solution over a space of solutions with a specified magnetic field strength was chosen. It is informative to view the objective function space over this surface. The logarithm of the objective function and the raw objective function are depicted in Figure 33. Because of the symmetry of the magentic field generated by a magnetic dipole, there are two solutions. There is a low-value band that circles around the curve that connects these two values. For a point that is on the equatorial line where $z = 0$, the band becomes



(a) $\ln(f(\hat{\mathbf{s}})$ (b) $f(\hat{\mathbf{s}}$

Figure 33. Depiction of the objective function over a surface with B-field values of a specific amplitude. The true value is $\mathbf{s} = [-0.02, 0.07, 0.015]$. It is possible to see the effect of the symmetric solution on the back side of the surface.

solid representing the circular symmetry that occurs at that point. If a vector $\mathbf{r}$ is made from the position of the manget to the sensor location, this occurs where the magnetic moment $\mathbf{m} \perp \mathbf{r}$, i.e. $\mathbf{m} \cdot \mathbf{r} = 0$. As the sensor moves increasingly far from this position, the two solutions become increasingly distinct and the bands connecting these two points become less pronounced.

It becomes possible to create an efficient way of making an initial estimate by sampling the data and taking the amplitude of the B-field of that sample and finding a directional vector to find the closest match. By considering the problem in the spherical vector coordinate, it is possible to exploit some of the properties of the magnetic field measurements to find an estimate of ϕ, reducing the problem to an optimization of a single parameter θ. To do this, it is necessary to convert the B-field measurements from a Cartesian vector field to a spherical vector field. This conversion is dependent upon the sensor position, so the conversion must be done on an estimated sensor position. The steps for performing this optimization over θ is presented here.

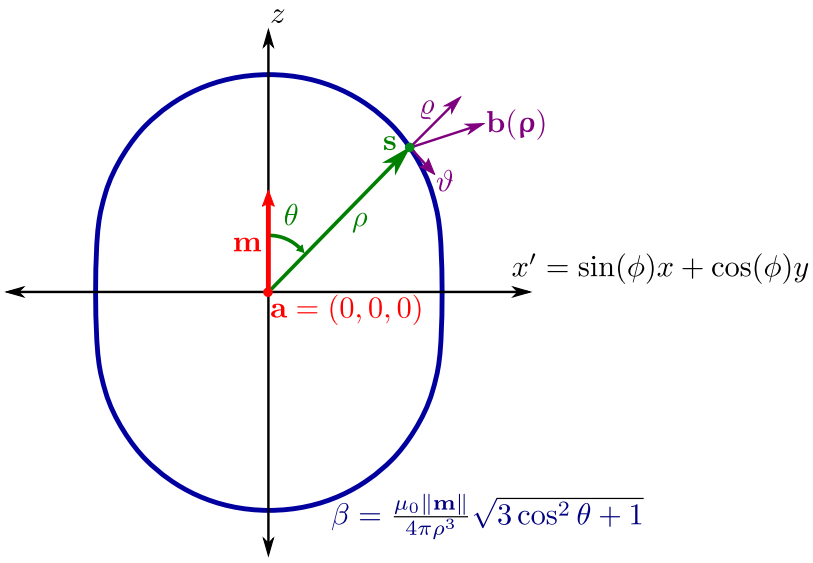


Figure 34. Depiction of the curve with constant field strength. The magnetic moment $\mathbf{m}$ is depicted as is the spherical representation of the magnetic field $\mathbf{b}(\boldsymbol{\rho})$ as measured at the sensor $\mathbf{s}$. Because the magnet is assumed to be at the origin, $\mathbf{s} = \boldsymbol{\rho}$

8.4.1 Description of the Method

The method being applied here is to take one sampled value of the magnetic field for a particular sensor for a particular magnet position. Using this information, try to identify an initial estimate of the sensor position. This reqirues some way of narrowing down the position of the sensor efficiently. This is easier to identify in spherical vector fields. The method generally follows the following steps.

Let β represent the strength of the B-field measured by a particular sensor. The space over which that sensor could be located forms a round, convex surface. The position of the sensor on this surface is identified by the orientation of the magnetic field. The solutions are non-unique based on the symmetry properties described in Section 8.2.1.

Because in spherical coordinates, the azimuthal-angle vector component $\varphi = 0$, it becomes simple to estimate ϕ by choosing an azimuthal direction to align with the vector field measurement. This constrains the sensor position to fall somewhere on a closed curve as depicted in Figure 34.

Finding the position on the curve can be found by performing an optimziation to find the polar angle θ. The following subsections outline the process to accomplish this.

8.4.2 Analysis in a Spherical Vector Field

Rather than using vectors in a Cartesian vector field, it is convenient in this problem to operate in a spherical vector field space. This utilizes the spherical vector representation of the B-field.

$$\mathbf{b}(\boldsymbol{\rho}) = \frac{\mu_0}{4\pi}\frac{\|\mathbf{m}\|}{\rho^3}\begin{bmatrix} 2\cos\theta \\ \sin\theta \\ 0 \end{bmatrix} \tag{160}$$

Because of the assumption that the magnet is centered at the origin with a moment oriented in the positive z direction, this may require additional rotation of the measured values and estimates of sensor positions estimated from this equation may require shifting the sensor positions.

Note that the $\hat{\phi}$ term φ is zero because magnetic fields are radial with respect to the azimuthal angle. The magnetic moment is in the z direction, so aligning the x, y portions of the B-field identifies the calculation of ϕ.

$$\phi_i = \arctan\frac{\left[\boldsymbol{\Xi}\mathbf{b}_i\right]_y}{\left[\boldsymbol{\Xi}\mathbf{b}_i\right]_x} \tag{161}$$

In practice, an estimate is taken by sampling a particular measurement sample with the index depicted by k.

Rather than trying to directly find a radius vector, the problem is reframed to find the strength of the magnetic field and use that value to derive the radius ρ.

$$\beta = \|\mathbf{B}(\boldsymbol{\rho}, \mathbf{a}, \mathbf{m})\| = \frac{\mu_0}{4\pi}\frac{\|\mathbf{m}\|}{\rho^3}\sqrt{4\cos^2\theta + \sin^2\theta} \tag{162}$$

It is straightforward to estimate the value of β from a particular sample k by finding the magnitude of the sampled B-field measurement.

$$\beta_{0,i} = \|\mathbf{b}_i[k]\| \tag{163}$$

Given a value of β, it becomes possible to express the value of $\rho = \|\mathbf{r}\|$ as a function of θ. For the optimization, the β value selected uses β_0.

$$\rho(\theta) = \left[\frac{\mu_0}{4\pi}\frac{\|\mathbf{m}\|}{\beta}\sqrt{3\cos^2\theta + 1}\right]^{\frac{1}{3}} \tag{164}$$

The expression of the B-field as expressed in a spherical vector field as a function of β and θ can be calculated by substituting $\rho(\theta)$ into Equation 8.2.1.

$$\mathbf{b}(\boldsymbol{\rho}) = \frac{\beta}{\sqrt{3\cos^2\theta + 1}}\left(2\cos\theta\hat{\rho} + \sin\theta\hat{\theta}\right) \tag{165}$$

It is possible to create an optimization problem to find an estimate of the polar angle θ using β_0 and ϕ_0, since its relationship is not immediately apparent from the other equations. The analysis is performed on the spherical vector field vectors, so the measurements (provided in Cartesian values) must be converted. The use of the estimation from (148) assumes the magnetic moment is oriented in the positive z direction. If this is not the case, an additional rotation of the sensor measurements is performed before the calculation with an inverse rotation after the calculation. It also assumes the magnet is oriented at the origin. If this is not true, it is necessary to take some steps to permit these calculations.

The process is done by finding a rotation that rotates the magnetic moment $\mathbf{m}$ to be pointed in the positive direction. It was demonstrated in Chapter 7 that there is a class of available rotations that rotate one vector to point in the direction of another vector in $\mathbb{R}^3$. One way of thinking about this is that there can be some spin about the destination vector that cannot be accounted for from a single pair of vectors.

In this case, however, this is not a concern since a magnetic dipole has a rotational symmetry.

This means that a rotation quaternion to adjust for the magnetic moment, $\mathbf{q_m}$ can be expressed by the following quaternion representation.

$$\theta = \arccos\left(\frac{\mathbf{m}}{\|\mathbf{m}\|} \cdot \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^\intercal\right) \tag{166}$$

$$\mathbf{v} = \frac{1}{\sin(\theta)}\left(\frac{\mathbf{m}}{\|\mathbf{m}\|} \times \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^\intercal\right) \tag{167}$$

$$\mathbf{q_m} = \left(\cos\left(\frac{\theta}{2}\right), \sin\left(\frac{\theta}{2}\right)\mathbf{v}\right) \tag{168}$$

Essentially, this represents a context shift into a new reference frame. For notation, a matrix $\mathbf{Q_m}$ corresponds to the equivalent rotation matrix. This will be used symbolically to provide notational consistency.

To be able to perform calculations in this space, it is necessary to apply this same rotation to the sensor measurements. From the previous chapter, the sensors were all placed into a common reference frame, but this reference frame is likely distinct from the true frame of reference of calculations. To account for this, it is assumed that there is some known rotation, represented by the matrix $\mathbf{\Xi}$, that rotates sensor measurements from the common sensor reference frame into the physical reference frame as specified by the magnetic positions and the magnetic moment orientation. Currently $\mathbf{\Xi}$ remains unknown, but methods to find this value will be presented in later chapters. Thus, the expression for the sensor measurement in the frame of reference of the aligned magnetic moment, denoted $\mathbf{b}_{\text{mag}}$ is given by

$$\mathbf{b}_{\text{mag}} = \mathbf{Q_m}\mathbf{\Xi}\mathbf{b}_i. \tag{169}$$

To perform the optimization, the coordinates of the sensor position are specified

as $\boldsymbol{\rho}_0 = [\rho(\theta, \beta_0), \theta, \phi_0]$ where θ is the variable under optimization, $\rho(\theta, \beta_0)$ is the radius as a function of the angle θ and the estimated field strength, and ϕ_0 is the estimate of the azimuth angle. It is actually unnecessary to calculate ρ as part of the optimization. It is required, however, when trying to convert the spherical representation to Cartesian coordinate positions representing sensor positions.

Conversion of the vector $\mathbf{b}[k]$ to the spherical vector space is calculated using a multiplication with another rotation matrix to align the magnetic field vectors into the spherical reference frame. This rotation matrix is denoted $\mathbf{P}(\boldsymbol{\rho}_0)$.

$$\mathbf{b}_i(\boldsymbol{\rho}_0) = \begin{bmatrix} \varrho[k] \\ \vartheta[k] \\ \varphi[k] \end{bmatrix} = \mathbf{P}(\boldsymbol{\rho}_0)\mathbf{Q}_\mathbf{m}\boldsymbol{\Xi}\mathbf{b}_i[k] \tag{170}$$

$$\mathbf{P}(\boldsymbol{\rho}_0) = \begin{bmatrix} \sin\theta\cos\phi & \sin\theta\sin\phi & \cos\theta \\ \cos\theta\cos\phi & \cos\theta\sin\phi & -\sin\theta \\ -\sin\phi & \cos\phi & 0 \end{bmatrix} \tag{171}$$

This rotation matrix is a function of the unknown value θ, so the spherical vector values must be calculated as part of the calculation of the objective function.

It is possible to find a quaternion $\mathbf{q}_\mathbf{P}(\theta, \phi)$ that can do the same rotation by using the matrix-to-quaternion equations from Section 7.2. This expression is based on the first conversion form using q_w as the denominator term.

$$\mathbf{q}_\mathbf{P} = \begin{bmatrix} \frac{\sqrt{\sin(\theta+\phi)+1}}{2} \\ \frac{\sin(\theta)+\cos(\phi)}{2\sqrt{\sin(\theta,\phi)+1}} \\ \frac{\cos(\theta)+\sin(\phi)}{2\sqrt{\sin(\theta,\phi)+1}} \\ \frac{\cos(\theta+\phi)}{2\sqrt{\sin(\theta,\phi)+1}} \end{bmatrix} \tag{172}$$

This form does have a shortcoming, though, because there is a numerical instability for $\theta + \phi \approx -\frac{\pi}{2} + n2\pi$. These are removable singularities. A formal proof can

be made, but a simple, informal argument is readily available. Rotation matrices convert to unit quaternions so each individual element must be bounded, and the set of equations is a valid equation for all non-singular points. Thus there must be a converging sequence that converges to the correct value at the point of instability.

To avoid the numerical problems that arise, it is possible to use a complementary form. This form uses the fourth conversion form that uses q_z as the denominator term.

$$\mathbf{q_P} = \begin{bmatrix} \frac{\cos(\theta+\phi)}{2\sqrt{1-\sin(\theta,\phi)}} \\ \frac{\cos(\theta)-\sin(\phi)}{2\sqrt{1-\sin(\theta,\phi)}} \\ \frac{-\sin(\theta)+\cos(\phi)}{2\sqrt{1-\sin(\theta,\phi)}} \\ \frac{\sqrt{1-\sin(\theta+\phi)}}{2} \end{bmatrix} \tag{173}$$

This form also has a removable singularity and point of instability, but this one is at $\theta + \phi \approx \frac{\pi}{2} + n2\pi$. This set of values represents the opposite equivalence class of angles (different by an integer multiple of π), so one of the sets of equations will always work. For brevity in notation, the sum of angles will be evaluated by an equivalence class representation, i.e. the equivalent angle on $[-\pi, \pi]$. A strategy of selection is when $\theta + \phi \in [0, \pi)$, use the first equation. When $\theta + \phi \in [-\pi, 0)$, use the second equation. As a reminder, $\mathbf{q} = -\mathbf{q}$, but only one representation is presented for these equations.

8.4.3 Estimate Optimization Description

The objective function is an L^2 measure of the difference between the spherical vector field values. This is equivalent to performing the same operation for vectors described in a Cartesian frame of reference.

$$f_{\text{sfast}} = \|\mathbf{b}_{\text{est}}(\boldsymbol{\rho}_0) - \mathbf{b}_i(\boldsymbol{\rho}_0)\|^2 \tag{174}$$

The Jacobian is calculated by means of the following equations. This Jacobian happens to be a derivative since it is the derivative of a differentiable scalar-valued function with respect to a scalar variable, but it is calculated using matrix calculations so the term Jacobian will still be used.

$$\frac{\partial f_{\text{sfast}}}{\partial \theta} = 2\left(\mathbf{b}_{\text{est}}(\boldsymbol{\rho}_0) - \mathbf{b}_i(\boldsymbol{\rho}_0)\right)^{\intercal}\left(\frac{\partial \mathbf{b}_{\text{est}}(\boldsymbol{\rho}_0)}{\partial \theta} - \frac{\partial \mathbf{b}_i(\boldsymbol{\rho}_0)}{\partial \theta}\right) \tag{175}$$

$$\frac{\partial \mathbf{b}_{\text{est}}(\boldsymbol{\rho}_0)}{\partial \theta} = \frac{2\beta}{\sqrt{3\cos^2\theta + 1}^3}\begin{bmatrix} -\sin\theta \\ 2\cos\theta \\ 0 \end{bmatrix} \tag{176}$$

$$\frac{\partial \mathbf{b}_i(\boldsymbol{\rho}_0)}{\partial \theta} = \frac{\partial \mathbf{P}(\boldsymbol{\rho}_0)}{\partial \theta}\boldsymbol{\Xi}\mathbf{b}_i[k] \tag{177}$$

$$\frac{\partial \mathbf{P}(\boldsymbol{\rho})}{\partial \theta} = \begin{bmatrix} \cos\theta\cos\phi & \cos\theta\sin\phi & -\sin\theta \\ -\sin\theta\cos\phi & -\sin\theta\sin\phi & -\cos\theta \\ 0 & 0 & 0 \end{bmatrix} \tag{178}$$

Once the optimization is complete, it is possible to find the sensor position by using β_0 and θ to calculate ρ and then by using θ and ϕ to specify the position in spherical coordinates that can be translated into Cartesian coordinates.

It may be beneficial (including, in particular) the next block, to continue to use values in β, θ, and ϕ. If an alignment rotation was required as an initial step, it is necessary to apply the inverse transformation $\mathbf{Q}^{\intercal}$. While it may seem that it is only necessary to apply this rotation to the rotational components, it must be remembered that β_0 is also a function of θ. Assuming that the rotation from θ_1 and ϕ_1 is made to θ_2 and ϕ_2 using the rotation $\mathbf{Q_m}$, the corresponding calculation of β_2 from β_1 is given by

$$\beta_2 = \beta_1 \frac{\sqrt{3\cos^2(\theta_2) + 1}}{\sqrt{3\cos^2(\theta_1) + 1}} \tag{179}$$

The meaning of this calculation is quite interesting. The value of β is representative of the strength of a magnetic field at a point. This expression identifies how strongly the magnetic moment is acting on the sensor as the relative angle of the magnetic moment changes.

There are many different frames of reference to be considered in the methods described here and it is important to ensure that any information being passed between sections is interpreted in the correct frame of reference.

To clarify the process, the algorithm is outlined in a pseudocode block in Algorithm 1.

Data: $\boldsymbol{\Xi}, \mathbf{b}_i$
Result: Estimate of θ
begin
  if *the magnetic moment* $\mathbf{m}$ *is not oriented in* $+z$ **then**
    rotate sensor measurement by $\mathbf{Q}_{\mathbf{m}}$, a rotation mapping $\mathbf{m}$ to +z
  end
  Select a random index k of sample $\mathbf{b}_{\mathrm{s}} = (\boldsymbol{\Xi}\mathbf{b}_i)[k]$
  Find estimate of β as $\beta_0 = \|\mathbf{b}_i[k]\|$
  Find estimate of ϕ as $\phi_0 = \arctan \frac{b_{i,y}}{b_{i,x}}$
  Optimize for θ using the objective function f_{sfast} and Jacobian $\frac{f_{\mathrm{sfast}}}{\partial\theta}$
  Calculate the Cartesian coordinates of the sensor position
  if *the magnetic moment* $\mathbf{m}$ *is not oriented in* $+z$ **then**
    rotate sensor position estimate by $\mathbf{Q}_{\mathbf{m}}^{\mathrm{T}}$
  end
  if *the magnet is not located at true origin* **then**
    add the position of the magnet to the sensor position estimate
  end
end

Algorithm 1: Algorithm outlining the way to make a quick estimate of the sensor position.

This optimization is clearly not convex, since it is periodic. This function does, however, exhibit the property that all local minima are global minima and they are periodic with period π. This is because of the symmetry of the problem. Additional information may be required to identify which is the correct sensor position.

While this initial condition estimate process requires the execution of an optimization requiring multiple evaluations of f_{sfast}, this optimization can be performed much more efficiently than the original objective function f_{direct} since it only requires working with a single sample vector and it optimizes over a single variable. This optimization is also well-behaved when compared with the direct function.

The accuracy of this method is a function of the noise, but it is able to find the optimal solution (up to the optimization tolerance) as possible given a single vector sample with the caveat that it is indeterminate if the radius vector is orthogonal to the magnetic moment.

8.5 Spherical Optimization Method

It is possible to extend the initial condition method to perform a complete optimization in the spherical vector space rather than in the Cartesian space. In this optimization, the values to discover are β, θ, ϕ. In this case, the hat $\hat{}$ is omitted from designating estimates to avoid confusion with the unit norm directions for spherical vector representations.

The objective function is the same as in the fast estimate case, though the calculation of the $\mathbf{b}_{\text{est}}$ is calculated directly using β instead of ρ. Also, this method includes a sum over each of the samples. To emphasize that these measurements are in a spherical reference frame, the argument variable will be denoted $\boldsymbol{\varrho} = [\beta, \theta, \phi]$ and because of this $\mathbf{b}_i[k]$ will be denoted $\mathbf{b}_{i,k}$.

$$f_{\text{sfull}}(\boldsymbol{\varrho}) = \sum_k \|\mathbf{b}_{\text{est}}(\boldsymbol{\varrho}) - \mathbf{b}_{i,k}(\boldsymbol{\varrho})\|^2 \tag{180}$$

The calculation of this Jacobian is a generalization of the one presented in the quick estimate version.

$$\frac{\partial f_{\text{sfull}}(\boldsymbol{\varrho})}{\partial \boldsymbol{\varrho}} = 2\sum_k (\mathbf{b}_{\text{est}}(\boldsymbol{\varrho}) - \mathbf{b}_{i,k}(\boldsymbol{\varrho}))^{\intercal} \left(\frac{\partial \mathbf{b}_{\text{est}}(\boldsymbol{\varrho})}{\partial \boldsymbol{\varrho}} + \frac{\partial \mathbf{b}_{i,k}(\boldsymbol{\varrho})}{\partial \boldsymbol{\varrho}} \right) \tag{181}$$

$$\frac{\partial \mathbf{b}_{\text{est}}(\boldsymbol{\varrho})}{\partial \boldsymbol{\varrho}} = \frac{1}{\sqrt{3\cos^2\theta + 1}} \begin{bmatrix} 2\cos\theta & \frac{-2\beta\sin\theta}{3\cos^2\theta+1} & 0 \\ \sin\theta & \frac{4\beta\cos\theta}{3\cos^2\theta+1} & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{182}$$

$$\frac{\partial \mathbf{b}_i(\boldsymbol{\varrho})}{\partial \boldsymbol{\varrho}} = \frac{\partial \mathbf{P}(\boldsymbol{\varrho})}{\partial \boldsymbol{\varrho}} \boldsymbol{\Xi} \mathbf{b}_i[k] \tag{183}$$

The calculation of the Jacobian is a little more complicated in notation as $\frac{\partial \mathbf{P}(\boldsymbol{\varrho})}{\partial \boldsymbol{\varrho}}$ represents the derivative of a matrix with respect to a vector which results in a four-dimensional tensor. It is not a problem so long as the dot product is taken across the proper dimensions.

$$\frac{\partial \mathbf{P}(\boldsymbol{\varrho})}{\partial \boldsymbol{\varrho}} = \begin{bmatrix} \frac{\partial \mathbf{P}(\boldsymbol{\varrho})}{\partial \beta} & \frac{\partial \mathbf{P}(\boldsymbol{\varrho})}{\partial \theta} & \frac{\partial \mathbf{P}(\boldsymbol{\varrho})}{\partial \phi} \end{bmatrix} \tag{184}$$

The matrix slices of the tensor are given by the following. Here $\mathbf{0}$ represents the $\mathbb{R}^{3\times 3}$ zero matrix.

$$\frac{\partial \mathbf{P}(\boldsymbol{\varrho})}{\partial \beta} = \mathbf{0} \tag{185}$$

$$\frac{\partial \mathbf{P}(\boldsymbol{\varrho})}{\partial \theta} = \begin{bmatrix} \cos\theta\cos\phi & \cos\theta\sin\phi & -\sin\theta \\ -\sin\theta & -\sin\theta\sin\phi & -\cos\theta \\ 0 & 0 & 0 \end{bmatrix} \tag{186}$$

$$\frac{\partial \mathbf{P}(\boldsymbol{\varrho})}{\partial \phi} = \begin{bmatrix} -\sin\theta\sin\phi & \sin\theta\cos\phi & 0 \\ -\cos\theta\sin\phi & \cos\theta\cos\phi & 0 \\ -\cos\phi & -\sin\phi & 0 \end{bmatrix} \tag{187}$$

Again some work can be done to arrange this to be calculated more efficiently. Once again, this will be divided by the number of entries for normalization purposes.

$$f_{\text{sfull}}(\boldsymbol{\varrho}) = \frac{1}{K}\sum_{k} \left\| \mathbf{b}_{\text{est}}(\boldsymbol{\varrho}) - \mathbf{P}\mathbf{Q}_{\mathbf{m}}\boldsymbol{\Xi}\mathbf{b}_{i,k} \right\|^2 \tag{188}$$

$$= \frac{1}{K}\sum_{k} \left\| \boldsymbol{\Xi}^{\intercal}\mathbf{Q}_{\mathbf{m}}^{\intercal}\mathbf{P}^{\intercal}\mathbf{b}_{\text{est}} - \mathbf{b}_{i,k} \right\|^2 \tag{189}$$

With this, it becomes possible to apply all the rotations to the single estimate value on each evaluation of the objective function rather than applying them all to the sampled measurements. For the Jacobian, it becomes possible to do even more simplification.

$$\frac{\partial \mathbf{b}_i(\boldsymbol{\varrho})}{\partial \boldsymbol{\varrho}} = 2\left[\mathbf{b}_{\text{est}}^{\intercal}\frac{\partial \mathbf{b}_{\text{est}}}{\partial \boldsymbol{\varrho}} - \overline{\mathbf{b}_{i,k}}\mathbf{Q}_{\mathbf{m}}^{\intercal}\frac{\partial}{\partial \boldsymbol{\varrho}}\mathbf{P}^{\intercal}\mathbf{b}_{\text{est}} \right] \tag{190}$$

The last derivative term in this expression is again a tensor dot product, just as before, but now it is being applied to the B-field estimate instead of the input data. This results in a matrix with columns given by the expressions below. Each entry represents a column vector.

$$\frac{\partial}{\partial \boldsymbol{\varrho}}\mathbf{P}^{\intercal}\mathbf{b}_{\text{est}} = \left[\mathbf{P}^{\intercal}\tfrac{\mathbf{b}_{\text{est}}}{\partial\beta} + \tfrac{\partial \mathbf{P}^{\intercal}}{\partial\beta}\mathbf{b}_{\text{est}},\ \ \mathbf{P}^{\intercal}\tfrac{\mathbf{b}_{\text{est}}}{\partial\theta} + \tfrac{\partial \mathbf{P}^{\intercal}}{\partial\theta}\mathbf{b}_{\text{est}},\ \ \mathbf{P}^{\intercal}\tfrac{\mathbf{b}_{\text{est}}}{\partial\phi} + \tfrac{\partial \mathbf{P}^{\intercal}}{\partial\phi}\mathbf{b}_{\text{est}} \right] \tag{191}$$

This construction means that again all the calculations can be performed in one step and only the average of the estimates is required in the iteration.

The initial estimate of the sensor position can be found using the fast spherical method here as well.

8.6 Results

To be able to compare the results, a common set of parameters were provided. The magnet being used in the estimation is positioned at $\mathbf{a} = (0, 0, 0)$ with a magnetic moment strength that is calculated from a known magnet typical of the one that is used for tracking, a D21B-N52: $^1/_8$ in diameter with a height of $^1/_{16}$in and a

$\mathrm{Br}_{\mathrm{Max}} = 14800\mathrm{G}$. The strength of the magnetic moment is given at $14.80\ {}^{\mathrm{mA}}/_{\mathrm{m}^2}$ using the calculation from Equation 145. Additional testing was performed with differing values of $\mathbf{a}$ to ensure that estimates correctly account for shifts in the magnet position.

The true sensor positions were selected to be representative of normal positions that would be used in the physical headset. These values are specified as $\mathbf{s}_1 = (1, -5.5, 1)$ cm, $\mathbf{s}_2 = (1, 5.5, 1)$ cm, $\mathbf{s}_3 = (-1, -7, 2)$ cm, and $\mathbf{s}_4 = (-1, 7, 2)$ cm. This provides two sensors that are slightly closer and two that are slightly further away. It is also noted that the z value is positive in both sets. It is assumed that, for the calibration method, the headset would be placed on a table and a magnetic position tool would be placed on the table putting the magnet being tracked below the sensors. When running the algorithms, any values with negative z components are reflected in the estimate.

While it is not particularly relevant at this stage of analysis, an additional common random orientation $\boldsymbol{\Xi}$ was specified for the sensors. This represents that the sensor measurements are only specified up to some common orientation from the earlier steps and it is necessary that the orientation be aligned. This was done to ensure that the algorithms were successful in adjusting for some known orientation adjustment. Later chapters will discuss how to find this sensor orientation.

Because the distance to the sensors varies and the strength of the field is a function of the relative orientation of the magnetic moment with respect to the direction of the sensor, the definition of the noise is a little bit more difficult to define consistently. For this analysis, it is assumed that the noise is statistically independent of the measurement strength. Because the strength of the field is proportional to the radius ρ, an estimate of the B-field strength is calculated by

$$\beta \sim \frac{\mu_0}{4\pi} \frac{\|\mathbf{m}\|}{\rho^3}. \tag{192}$$

A common radius estimate is used across all values and is approximated as 7cm,

so the standard deviation of the noise as stated is multiplied by 0.4315 G. This allows a common level of noise to be present across all different orientations of the magnetic moment and provides consistency across all the different magnetic sensors. It is important to consider the way this is calculated when interpreting the noise terms in the results.

8.6.1 Execution Time

For the test of the execution time, both methods make use of the same initial estimator. At this point, it is clear that the normal L^2 method is more efficient. A graph of the runtime comparing the two methods is in Figure 35. It takes approximately twice as long to run the optimization for the spherical method. This is due to the increased amount of calculation. It seems that, with a good initial guess, the two systems converge comparably well. Failing to provide a good initial guess, however, can lead to divergence, so it remains important to use the spherical estimator for finding good initial estimates.

The runtime is also clearly impacted by the noise present. This can be explained in part becauese the initial estimate becomes less effective with increasing levels of noise, but increased noise smooths out the gradient making convergence a little bit slower as well. Results of the runtime from simulations as a function of the number of values as parameterized by the amount of noise is in Figure 36.

While the spherical method is slower than the direct method, the rates are more consistent. From this view, it seems that the spherical method may be slightly better behaved than the direct method, but it appears that the overhead does limit the ability for it to compete with a normal L^2 norm across all the tested noise levels which cover all expected operating conditions.

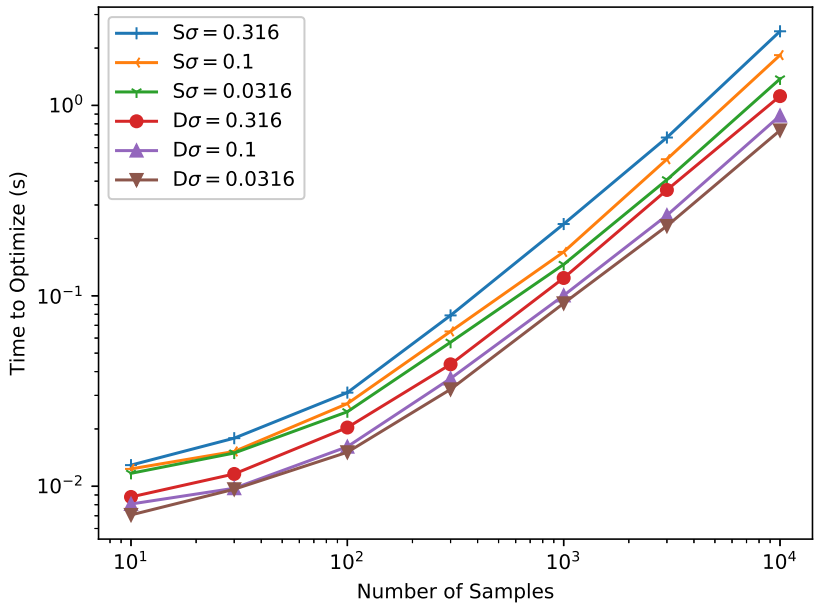
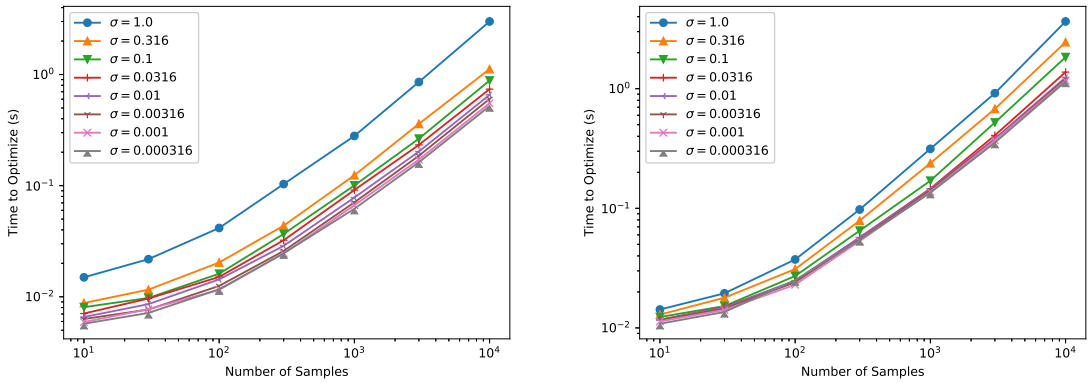


Figure 35. Comparison of the execution rate as a function of the number of samples for the spherical method (S) and the direct L^2 method (D) for noise values of $\sigma = 0.316,\ 0.1$, and 0.0316



(a) Direct L^2 method

(b) Spherical method

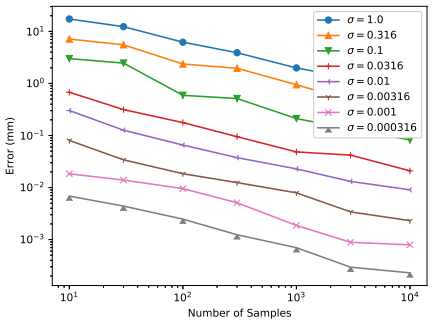
Figure 36. Depiction of the runtime as functions of the number of samples conditioned on the level of the noise in the samples

8.6.2 Accuracy

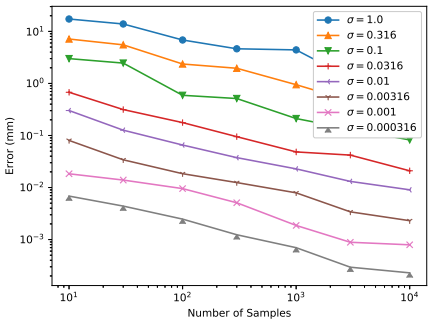
Accuracy is measured by comparing the L^2 norm of the distance between the true location value and the location estimate. Results of the experimental tests are in Figure 37. Estimates were taken from more than 100 simulations. It seems that the L^2 method is more likely to have performance problems for high amounts of noise. This can be seen for sensors 3 and 4 (the sensors farthest from the magnet) at the highest noise level indicating that that method seems to be a little bit more stable. Otherwise, the two methods seem to be indistinguishable.

What seems to be a more important consideration on the accuracy is the orientation of the magnetic moment when compared with the direction of the sensor. When the magnetic moment is orthogonal to the vector in the direction of the sensor, there is a large amount of uncertainty about the exact location of the magnet. This means that small deviations can lead to large errors. To show this, a scatter plot of the error measurements as a function of the angle between the magnetic moment and the vector pointing in the direction of the sensor is in Figure 38. For this experiment, the sensor under test was sensor 1 with a noise parameter of 0.01.

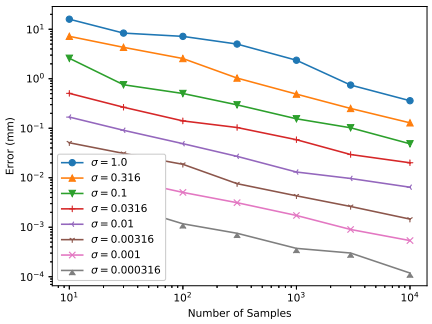
The error as the angle approaches perpendicular is approximately ten times larger than only ten degrees away from perpendicular. There is also a slight trend as the angle becomes parallel to further reduce the error. This is due to the fact that as the angle approaches perpendicular and with a constant radial distance, the strength of the signal is twice as strong as compared to when the signal is perpendicular since the values are related by $\sqrt{3\cos^2(\theta)+1}$. This means that, when trying to track a magnet, it is best to avoid a magnetic moment perpendicular to the vector from the magnet to the sensor positions. For sensor placement in principle, it is best if the sensors are not coplanar to ensure that some of the sensors are not in a perpendicular direction to the magnet under all circumstances. It is also beneficial if the magnetic moment is pointed in the direction of the sensor. Placing the sensors closer to the



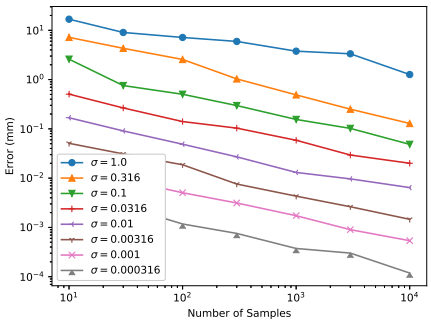
(a) Sensor 1, L^2



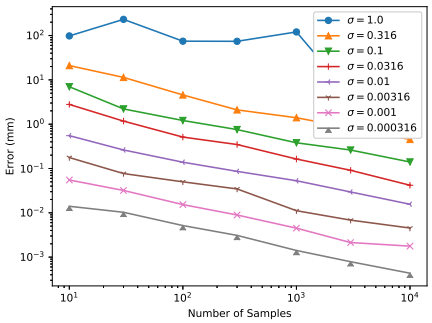
(b) Sensor 1, Spherical



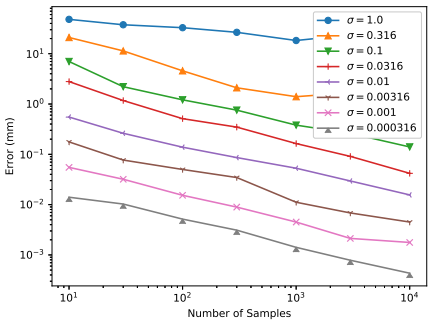
(c) Sensor 2, L^2



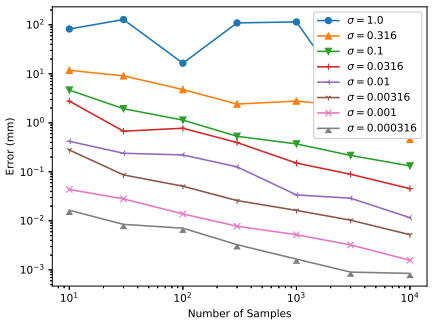
(d) Sensor 2, Spherical



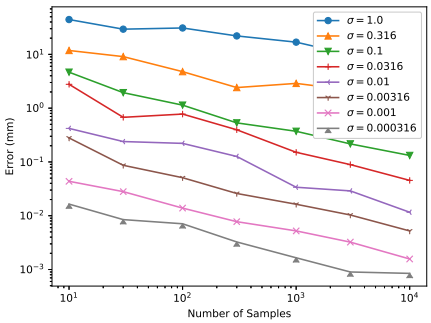
(e) Sensor 3, L^2



(f) Sensor 3, Spherical



(g) Sensor 4, L^2



(h) Sensor 4, Spherical

Figure 37. Comparison of the sensor estimate accuracy as a function of number of samples across each of the four sensors.

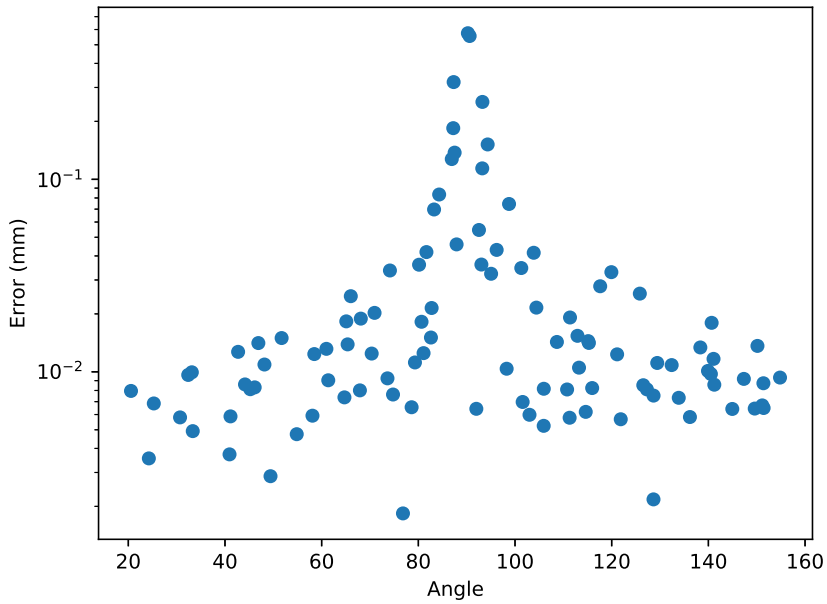


Figure 38. Comparison of the error as a function of the angle between the magnetic moment and the sensor position. The error for this test is $\sigma = 0.01$ and this test is for sensor 1.

magnet is also beneficial as it improves the SNR, though this is limited physically by usability requirements of the device.

CHAPTER IX

GLOBAL ORIENTATION

9.1 Aim

This stage of processing assumes that the measurements have been processed with a calibration and a relative orientation so that each of the magnetic sensors should exist in a common alignment. Additionally, the measurements from the fifth sensor are subtracted from the sensor measurements.

This stage attempts to reorient the sensor measurements so that they are in the same orientation as the physical space. To accomplish this, a magnet to track is placed in known orientations and in multiple positions and measurements are recorded by the sensors for each location. It is expected that the headset remains stationary during this process, so it is possible to sample the magnetic field measurements without the presence of a tracking magnet and the average sensor measurements can be subtracted from the measurements after processing them for calibration. This should be equivalent to removing measurements from the reference sensor in the absence of the tracking magnet.

The calculation in this section also makes the assumption that the sensor positions are known.

9.2 Optimization

The representation of the rotation to find is given by $\boldsymbol{\Xi}$. The sensors indexed by i are identified as $\mathbf{s}_i$, the magnet positions indexed by j are $\mathbf{a}_j$ with magnetic moments $\mathbf{m}_j$, and the samples are indexed by k. Rotations in this section are denoted $\boldsymbol{\Xi}$, but they are defined in the same way as $\mathbf{Q}$ from Chapter 7, and so the $\frac{\partial \boldsymbol{\Xi}}{\partial \psi}$, $\frac{\partial \boldsymbol{\Xi}}{\partial \theta}$, $\frac{\partial \boldsymbol{\Xi}}{\partial \phi}$ are the

same as the corresponding partial derivatives with respect to $\mathbf{Q}$.

The algorithms used to do this rotation are the same as the rotations described in Section 7.3 where the values are rotated to align with the estimates as described by the sensor positions, the magnetic position, and the magnetic moment. In this case, the Euclidean affine transform does not work. Because each configuration is expected to have multiple samples, this would require the rotation of each sample vector. A more efficient method is instead to calculate the value that rotates the estimate to map onto the data. This means that the rotation to be calculated is $\boldsymbol{\Xi}^{-1}$. With quaternions, rotations are expressed in terms of unit vectors so the inverse is equivalent to the complex conjugate.

By way of notation, $\mathbf{b}_{i,j,k}$ represents a measurement from sensor i, with magnet position $\mathbf{a}_j$ and magnetic moment $\mathbf{m}_j$, for sample k. The estimate $\mathbf{b}_{\text{est}(i,j)} := \mathbf{b}_{\text{est}}(\mathbf{s}_i, \mathbf{a}_j, \mathbf{m}_j)$. The value I represents the number of sensors, J represents the number of magnet configurations, and K is the total number of samples. While it is expected that K be constant, it could also be a function of i and j.

There are three possible ways of doing the following optimizations: using a single magnet configuration j and analyze over sensor positions $1 \leq i \leq I$, using a single sensor i and optimize over multiple magnet configurations $1 \leq j \leq J$, and using multiple sensor positions $1 \leq i \leq I$ and over multiple magnet configurations $1 \leq j \leq J$. To save some space, the third, most-general case is presented and the more limited cases can be inferred by selecting one value and taking I or J to be equal to one.

9.2.1 Central Angle Distance Rotation

$$f_{\text{GCAD}}(\boldsymbol{\Xi}^{-1}) = \frac{1}{IJK} \sum_{i,j,k} \left[\arccos \frac{\mathbf{b}_{i,j,k}^{\intercal} \boldsymbol{\Xi}^{-1} \mathbf{b}_{\text{est}(i,j)}}{\|\mathbf{b}_{i,j,k}\| \|\boldsymbol{\Xi}^{-1} \mathbf{b}_{\text{est}(i,j)}\|} \right]^2 \tag{193}$$

$$\xi_{i,j,k} = \frac{\mathbf{b}_{i,j,k}^{\intercal} \boldsymbol{\Xi}^{-1} \mathbf{b}_{\text{est}(i,j)}}{\|\mathbf{b}_{i,j,k}\| \|\boldsymbol{\Xi}^{-1} \mathbf{b}_{\text{est}(i,j)}\|} \tag{194}$$

Then the Jacobian with respect to the matrix $\boldsymbol{\Xi}^{-1}$ is specified by the following

calculation. The same trace calculations outlined in Section 7.3 apply here to get from the matrix-valued Jacobian on $\boldsymbol{\Xi}^{-1}$ to the Jacobian vector for ψ, θ, ϕ.

$$\frac{\partial f_{\text{CAD}}(\boldsymbol{\Xi}^{-1})}{\partial \boldsymbol{\Xi}^{-1}} = -\frac{2}{IJK} \sum_{i,j,k} \frac{\arccos \xi_{i,j,k}}{\sqrt{1-\xi_{i,j,k}}} \frac{\partial \xi_{i,j,k}}{\partial \boldsymbol{\Xi}^{-1}} \tag{195}$$

$$\frac{\partial \xi_{i,j,k}}{\partial \boldsymbol{\Xi}^{-1}} = \left[\frac{\mathbf{b}_{i,j,k} \mathbf{b}_{\text{est}(i,j)}^{\intercal}}{\|\mathbf{b}_{i,j,k}\| \|\boldsymbol{\Xi}^{-1} \mathbf{b}_{\text{est}(i,j)}\|} - \frac{\mathbf{b}_{\text{est}(i,j)}^{\intercal} \boldsymbol{\Xi}^{-1\intercal} \mathbf{b}_{i,j,k} \boldsymbol{\Xi}^{-1} \mathbf{b}_{\text{est}(i,j)} \mathbf{b}_{\text{est}(i,j)}^{\intercal}}{\|\mathbf{b}_{i,j,k}\| \|\boldsymbol{\Xi}^{-1} \mathbf{b}_{\text{est}(i,j)}\|^3} \right] \tag{196}$$

The same considerations about the numerical instability that were employed in the analogous rotational alignment from the common orientation section also applies in this case and a Taylor Series expansion near $\xi = 1$ is applied (see Section 7.3.1 for more information.)

9.2.2 Euclidean Rotation

The Euclidean objective function is a natural one that is used repeatedly, but with derivatives with respect to different values.

$$f_{\text{GL2}}(\boldsymbol{\Xi}^{-1}) = \frac{1}{IJK} \sum_{i,j,k} \left\| \boldsymbol{\Xi}^{-1} \mathbf{b}_{\text{est}(i,j)} - \mathbf{b}_{i,j,k} \right\|^2 \tag{197}$$

The Jacobian of the objective function is given by the following expression.

$$\frac{\partial f_{\text{GL2}}(\boldsymbol{\Xi}^{-1})}{\partial \boldsymbol{\Xi}^{-1}} = \frac{1}{IJK} \sum_{i,j,k} 2\mathbf{b}_{\text{est}(i,j)} \left[\mathbf{b}_{\text{est}(i,j)}^{\intercal} \boldsymbol{\Xi}^{-1\intercal} - \mathbf{b}_{i,j,k}^{\intercal} \right] \tag{198}$$

Conversion to a Jacobian vector of ψ, θ, ϕ is the same again.

Because the estimate $\mathbf{b}_{\text{est}}$ is constant across all k and using a little bit of algebra, it becomes possible to simplify the expression of the Jacobian.

$$\frac{\partial f_{\text{GL2}}(\boldsymbol{\Xi}^{-1})}{\partial \boldsymbol{\Xi}^{-1}} = \frac{-2}{IJ} \sum_{i,j} \overline{\mathbf{b}_{i,j,k}} \mathbf{b}_{\text{est}(i,j)}^{\intercal} \tag{199}$$

Here, $\overline{\mathbf{b}_{i,j,k}}$ represents the vector values averaged over the samples k.

9.2.3 Projected Euclidean Rotation

$$f_{\mathrm{GPL2}}(\boldsymbol{\Xi}^{-1}) = \frac{1}{IJK}\sum_{i,j,k}\left\|\frac{\boldsymbol{\Xi}^{-1}\mathbf{b}_{\mathrm{est}(i,j)}}{\|\mathbf{b}_{\mathrm{est}(i,j)}\|} - \frac{\mathbf{b}_{i,j,k}}{\|\mathbf{b}_{i,j,k}\|}\right\|^2 \tag{200}$$

The Jacobian is calculated using the following function.

$$\frac{\partial f_{\mathrm{GPL2}}(\boldsymbol{\Xi}^{-1})}{\partial \boldsymbol{\Xi}^{-1}} = \frac{2}{IJK}\sum_{i,j,k}\left(\frac{\mathbf{b}_{\mathrm{est}(i,j)}\mathbf{b}_{\mathrm{est}(i,j)}^{\mathsf{T}}\boldsymbol{\Xi}^{\mathsf{T}}}{\|\mathbf{b}_{\mathrm{est}(i,j)}\|^2} - \frac{\mathbf{b}_{\mathrm{est}(i,j)}\mathbf{b}_{i,j,k}^{\mathsf{T}}}{\|\mathbf{b}_{\mathrm{est}(i,j)}\|\|\mathbf{b}_{i,j,k}\|}\right) \tag{201}$$

The same simplification used in the normal Euclidean version applies here as well.

$$\frac{\partial f_{\mathrm{GL2}}(\boldsymbol{\Xi}^{-1})}{\partial \boldsymbol{\Xi}^{-1}} = \frac{-2}{IJ}\sum_{i,j}\overline{\left(\frac{\mathbf{b}_{i,j,k}}{\|\mathbf{b}_{i,j,k}\|}\right)}\frac{\mathbf{b}_{\mathrm{est}(i,j)}^{\mathsf{T}}}{\|\mathbf{b}_{\mathrm{est}(i,j)}\|} \tag{202}$$

In this calculation, the normalization is done on each of the samples k and the average is taken over all k after the normalization step to make each of the vectors unit vectors.

9.3 Initial Estimate

The initial estimate technique is applied as described in Section 7.4. It is assumed that all the sensors share a common rotation orientation. To ensure that the rotations do not overlap, the selection of measurements to use for the estimate should come from different magnetic sensor positions and/or different sensors because the rotation estimate requires that the available vectors about which to rotate do not describe the same set of vectors.

9.4 Evaluation

For the purpose of this analysis, the magnet positions are assumed to be on a flat plane (in the (x, y) plane) with the magnetic moment pointed normal to the plane (in the positive-z direction.) The magnet is moved 2cm in each of the positions $\pm x$ and $\pm y$ directions resulting in a set of five magnet positions. The initial magnet position is

defined to be the origin and the magnet positions and magnetic orientations describe the orientation for which the magnet and sensor positions are expected to be defined.

The sensor positions used in this experiment are the same as those used in the sensor position estimate and are typical of real headset positions. These values are $\mathbf{s}_1 = (1, -5.5, 1)$ cm, $\mathbf{s}_2 = (1, 5.5, 1)$ cm, $\mathbf{s}_3 = (-1, -7, 2)$ cm, and $\mathbf{s}_4 = (-1, 7, 2)$ cm. Measurements were simulated at each of the sensor positions under each of the specified magnetic position configurations.

Magnetic field measurements are generated using the model and the sensor measurements are reoriented by using some rotation matrix $\mathbf{T}$. Noise is added to the measurements. The algorithms are then used to try and identify the orientation. The evaluation metric used is the same one that is used in Chapter 7.

$$\mu(\boldsymbol{\Xi}^{-1}, \mathbf{T}) = \arccos\left(|\mathbf{q}_{\boldsymbol{\Xi}^{-1}} \cdot \mathbf{q}_{\mathbf{T}}|\right) \tag{203}$$

The three different rotational algorithms were used in experiments. In the experiments, CAD represents the central angle objective function, L2 represents the L^2 Euclidean norm objective function, and PL2 represents the projected L^2 Euclidean norm objective function.

In order to get a correct rotation, it is necessary to provide at least two different contexts for calculating the rotations in three-dimensional space, as is outlined by the two-vector method for estimating a rotation. Because the experiments being done for this analysis include different magnet positions as well as different sensor measurements, there are a few different options for which terms to use for optimization. This parameter is referred to as the data type of the simulation. The data types are specified as "Mag" representing sampling values under the five different magnet positions with measurements taken by sensor s_1, "Sensor" representing sampling values from the four different magnetic sensors with the magnet in the origin position, and "Both" representing sampling values from both the five different magnetic positions

and the four different sensor measurements. Because of this, "Sensor" has four sets of measurements, "Mag" has five, and "Both" has twenty. This must be considered when comparing the different results and computational time.

These experiments have the same ambiguity in the noise level as the sensor position estimator. In this case, the noise strength is specified by the same multiplier of 0.4315 G.

The number of samples used in these experiments comes from a list of sample numbers {3, 10, 30, 100, 300, 1000, 3000, 10000}. Because of the multiple measurements, only three samples from each configuration item are used and it provides useful results.

9.4.1 Execution Time

The time to execute the code ranges from just over 10ms for 3 samples

The first comparison is made between the different methods of calculation. These results are in Figure 39. From this plot, it is apparent that the CAD method is less efficient. This is the same result that was found when the analogous methods were applied to putting all the sensors into a common alignment. The L2 and PL2 methods, however, demonstrate a slight shift where for low amounts of data L2 is more efficient and with large amounts of data PL2 is more efficient. The reason for this is because of the scaling. The PL2 method demonstrates a natural scaling of the vectors, so consequently the vectors are scaled up to unity where standard L2 is not. The optimization stopping criterion was left constant at a gradient difference within $10^{-}8$, so the L2 method, which lacks the scaling, is required to find a tighter fit before meeting the stopping criterion. The central angle distance also has its own natural scaling. A better direct comparison could be made by introducing a scaling term to the optimization, but this scaling is proportional to the vector strength which is about equal to the variance multiplier which was set to 0.4315, so the difference is

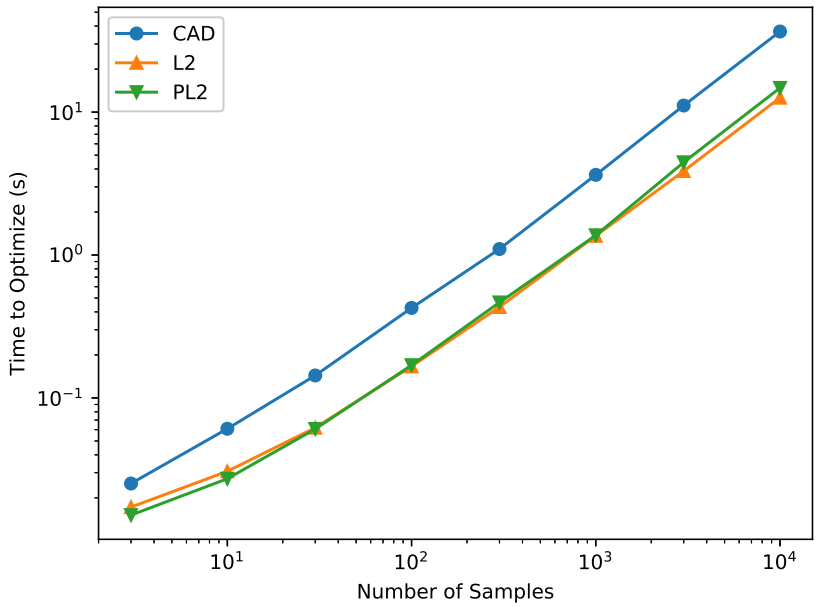


Figure 39. Comparison of the runtime across the three different algorithms. CAD represents the central-angle distance objective function, L2 is the Euclidean objective function, and PL2 is the projected Euclidean objective function.

within one order of magnitude.

Another way to compare the different methods is based on how efficient the optimizations are as a function of the type of optimization. These results are displayed in Figure 40

The calculation of the system using magnet position data is about 1.5 times slower than using the different sensor measurements. This is a little slower than the $5/4$ factor in the amount of data used. Using both sets of data is slower, as would be expected that is about and five times slower at 10^4 samples than using sensor data only. This is expected since it has five times more data and thus the optimization techniques seem to be a complexity of order $\mathcal{O}(N)$.

Another comparison is execution as a function of the noise. The execution time as a function of the noise is quite small with the difference in execution time between the largest and smallest amounts of noise are within a factor of two. These results

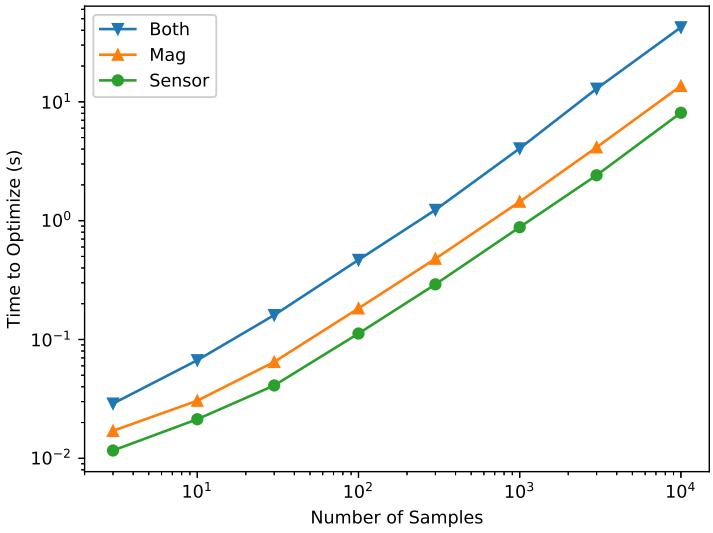


Figure 40. Comparison of the runtime across the three different data types. Mag uses measurements over the five magnet positions measured at sensor 1, Sensor uses measurements from the four different sensors with the magnet at the origin location, and Both uses both measurements taken from the five magnet positions across all four sensors.

are in Figure 41.

To be able to compare the runtime across different data types and different simulation types under different noise conditions. Comparisons of such simulations are tabulated in Figure 42.

From this, it seems that the PL2 optimizations are faster for the sensor-only method, the L2 is slightly faster for the magnet-only method, and they are approximately equal when both are considered. The CAD method takes approximately twice as long as the other methods in all cases. This is consistent across various noise strengths.

9.4.2 Accuracy

To determine the accuracy of the rotation, the rotation metric was used to compare the accuracy of the estimated rotation when compared with the true rotation under the various test conditions.

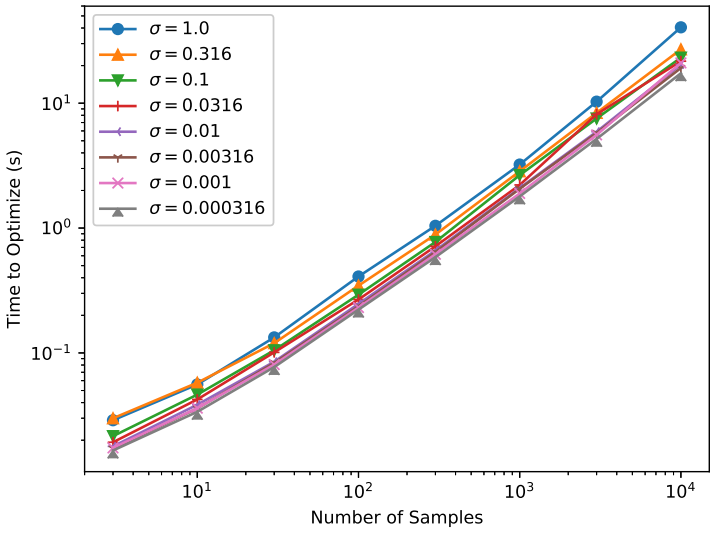
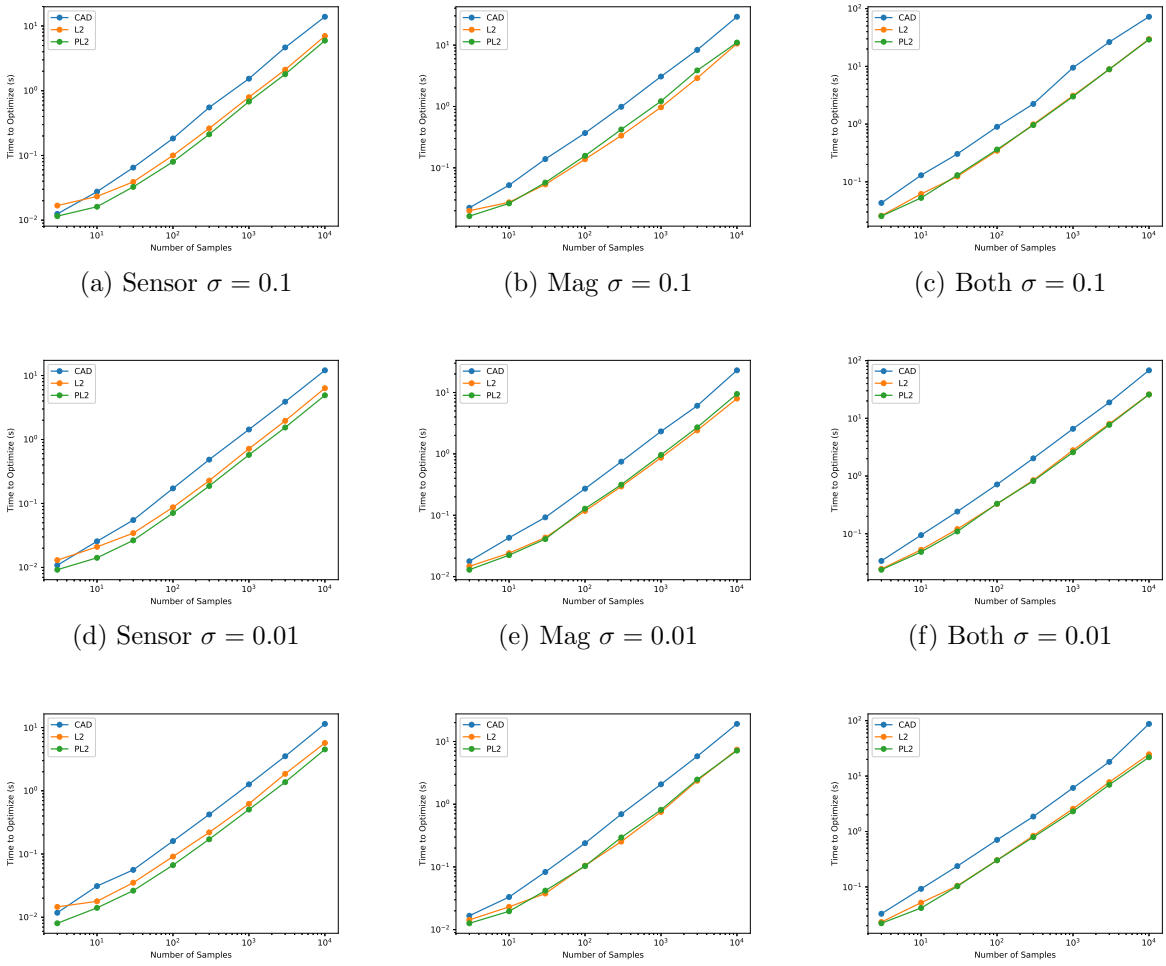


Figure 41. Comparison of the runtime based on the amount of noise in the system. The standard deviations σ are scaled by a constant value of 0.4315, a scaling term to roughly approximate the strength of the magnetic fields observed at the different sensors.

The first comparison checks the difference in accuracy between the different algorithms. The results are in Figure 43. The CAD method was the least accurate method and L2 was the most accurate. The difference in the accuracy between the L2 and the PL2 method is probably largely to do with the scaling as it interacts with the stopping criterion of the optimization. Because the sensor measurements are roughly 0.43 G, that multiple scaling applies to the objective function and thereby the Jacobian. This means that the stopping criterion is harder to achieve. This is why the optimization of the L2 method is so similar to the PL2 method when then PL2 method requires more computation.

The comparison of accuracy as a function of the different types of data measurements is provided in Figure 44. The interesting thing in this case is that, despite having slightly less data available, using data from multiple sensors for a single magnet position is more accurate than using measurements from a single sensor with different magnet positions. There is a risk in a practical system that there may be a



(a) Sensor $\sigma = 0.1$ (b) Mag $\sigma = 0.1$ (c) Both $\sigma = 0.1$

(d) Sensor $\sigma = 0.01$ (e) Mag $\sigma = 0.01$ (f) Both $\sigma = 0.01$

(g) Sensor $\sigma = 0.001$ (h) Mag $\sigma = 0.001$ (i) Both $\sigma = 0.001$

Figure 42. Comparison of the runtime of the different methods (distinguised by lines identified in the legends) with different data types (left to right) under various noise parameters (top to bottom.)

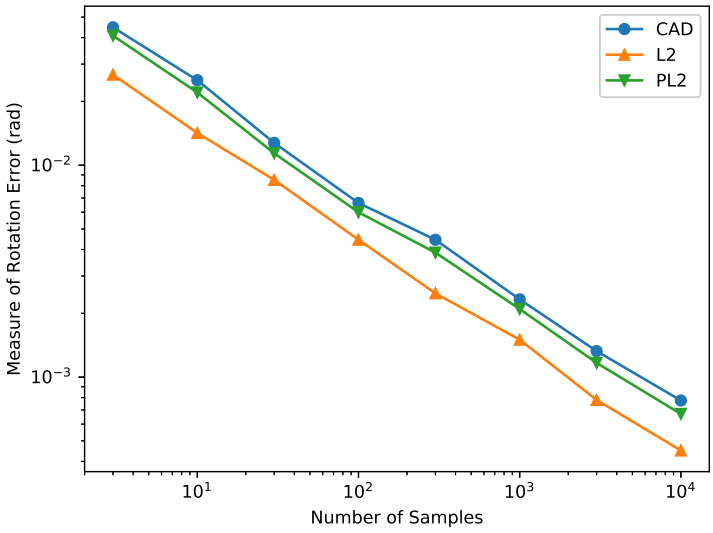


Figure 43. Comparison of the accuracy of the rotation estimate as parameterized by various algorithms. CAD is the central-angle distance objective function technique, L2 uses a Euclidean norm, and PL2 uses a projected Euclidean norm.

slight mismatch between different sensors, which could weaken this advantage, but it is something worth noting. Of course using all the data leads to improved accuracy.

The accuracy as a function of the noise is also found and is presented in Figure 45. As would be expected, the accuracy increases with more data. The tracking is roughly linear, though the lowest noise setting seems to be approaching a limit that is probably based on the stopping criteria of the optimizations. It is interesting to see that, even in the highest noise setting and with only three samples per configuration term, the results are a notable improvement over a random guess (a value of approximately 1 rad with this metric.)

To take a closer look at the accuracy under different noise estimates using the different algorithms, results using both magnet and sensor measurements were done using each of the three methods. Results under different noise conditions were compared. The results are in Figure 46. From this it can be seen that the L2 norm shows some inaccuracies for low amounts of noise while the other two techniques seem to

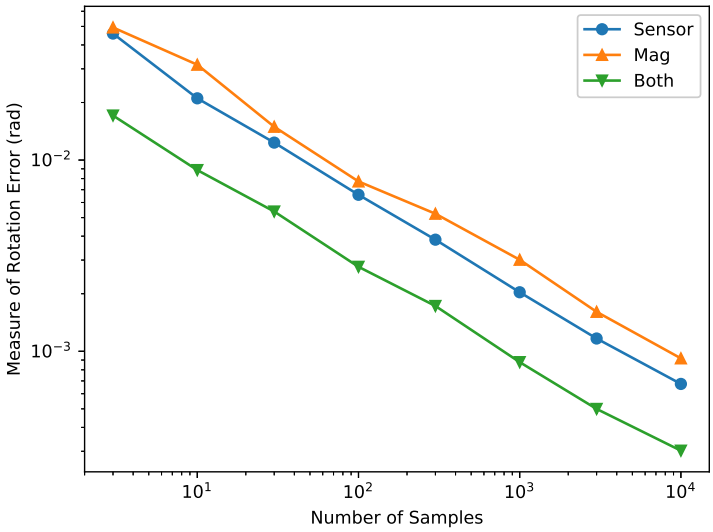


Figure 44. Comparison of the accuracy of the rotation estimate as parameterized by various data types. Mag represents using five magnet positions sampled at sensor 1, Sensor represents taking samples from all four sensors with a magnet at the origin, and Both uses data from all four sensors for all five magnet positions.

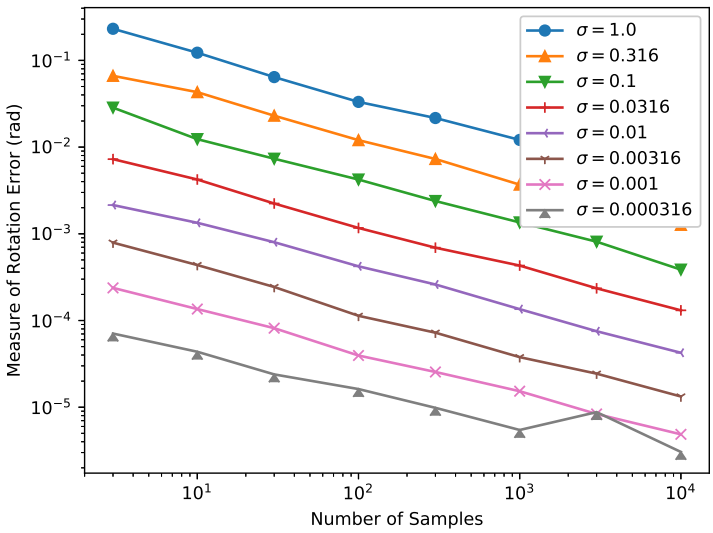
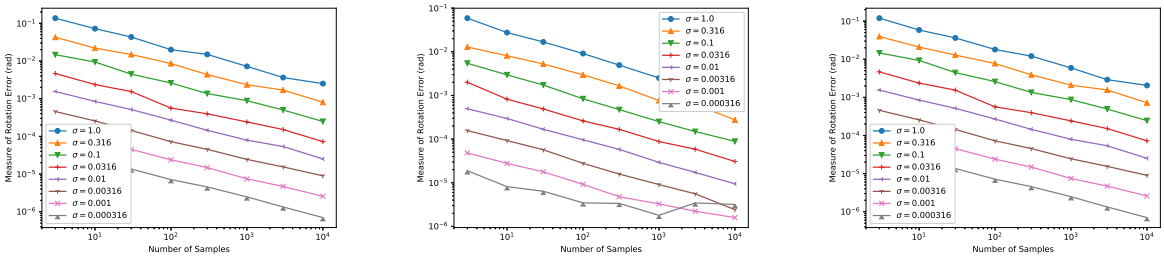


Figure 45. Comparison of the accuracy of the rotation estimate as parameterized by noise levels. The strength of the noise has a standard deviation of σ multiplied by a scaling term of 0.4315 to scale it to a value roughly proportionate to the signal strength at the sensors.



(a) CAD (b) L2 (c) PL2

Figure 46. Comparison of the accuracy of the different methods. CAD uses the central-angle distance norm, L2 uses the Euclidean norm, and PL2 uses the Euclidean norm of the projection. The noise levels had a standard deviation of σ scaled by a constant multiplier of 0.4315 to make the noise roughly proportional to the signal strength.

be a little bit more consistent in that region. These other techniques also exhibit the ability to outperform the L2 method for accuracy in low noise settings.

From these results, it seems that using the L2 norm gives efficient performance with a good degree of accuracy though there may be some times when using one of the more sophisticated methods may be beneficial.

CHAPTER X

COMBINATION POSITION AND ORIENTATION

10.1 Aim

It has been demonstrated that it is possible to, given a set of sensor positions and magnetic information, find a rotation to align sensor measurements to the orientation implied by the sensor positions and magnetic information. It has also been demonstrated that, given a rotation of the sensor measurements to the implied orientation, it is possible to accurately estimate the position of magnetic sensors.

It is necessary to now find both the orientation and position when both are simultaneously unknown.

10.2 Expectation-Maximization Solution

This method attempts to solve the problem by sequentially estimating the orientation and then the sensor position to try and find a local minimum. While both methods are able to converge to the correct solution, it is not immediately clear that, when one piece of information is not accurate, that the sequence of calculations generates a convergent sequence.

After running experiments of this optimization, the process proves to be divergent. In fact, even when accurate estimates of the orientation and sensor positions are provided, the alternating sequence leads to small errors which iteratively become worse. The values at successive iterations get further from the correct values and the objective function measurements also increase, which is important. The successive iterations continue to find local minima in the context of the parameterized system, but the minima move with each iteration causing the values to drift.

While each step does a good job of converging, the problem seems that the sequence of iterations does not make progress on the global problem. Using the objective functions as described in the previous chapters does not lead to convergence.

10.3 Sensor Estimation Alignment Method

Without being able to directly using the previously identified objective functions to lead to a convergent estimate, a new method is proposed. In order to lead to a solution for the general problem, the test needs to be adjusted to optimize according to an objective function which can assess the optimization for the problem in the context of the broader estimation.

To accomplish this, the metric is defined by performing multiple sensor position estimates though optimization and then building a metric based on how much those estimates align. This is expressed in terms of an optimization function to optimize for the rotation $\boldsymbol{\Xi}$. Under that rotation value, an optimization is run to estimate $\mathbf{s}_i$ as a function of each magnetic position and orientation $\mathbf{a}_j, \mathbf{a}_j$. This means that given each J known magnet positions and orientations, there is an estimate of the sensor position, denoted as $\hat{\mathbf{s}}_{i,j}$. It is possible to generate these estimates for all of the magnet-tracking sensors but, because they are all assumed to have a common orientation, they should all share a rotation $\boldsymbol{\Xi}$, so an estimation can be performed using only information for a single sensor.

To create a metric over which to optimize, a sum of L^2 measure between sensor position estimates is found.

$$f_{\text{SEAM}} = \sum_{j_1=0}^{J} \sum_{j_2=j_1}^{J} \|\hat{\mathbf{s}}_{i,j_1} - \hat{\mathbf{s}}_{i,j_2}\| \tag{204}$$

This represents the sum of the pairwise distances between the sensor estimates. This is graphically depicted in Figure 47. The magnetic sensor position estimates can be performed using any of the algorithms described in Chapter 8.

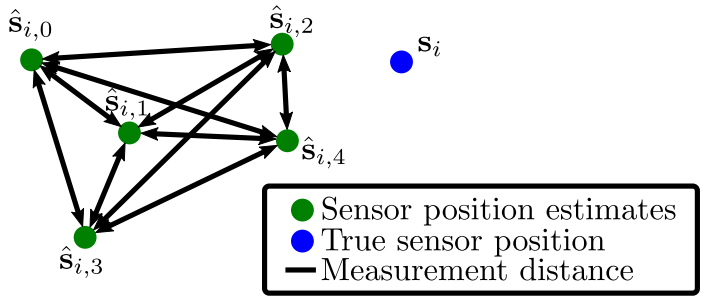


Figure 47. Depiction of the objective function. Under some rotation $\boldsymbol{\Xi}$, an estimate for a sensor can be made for each magnet configuration j. As $\boldsymbol{\Xi}$ approaches the accurate value, the sensor estimates approach the true position. This decreases the sum of the pairwise distances between the sensor estimates.

The processing of the estimates are done through an optimization algorithm, so it is not possible to calculate a Jacobian for this function. This requires using a method that requires no Jacobian, such as the Nelder-Mead algorithm. Not having access to a method for calculating the Jacobian requires more evaluations of the objective function, which is an important consideration.

An outline of the objective function algorithm is outlined here.

Data: $i, \mathbf{b}_{i,j,k}, \mathbf{a}_j, \mathbf{m}_j, \hat{\boldsymbol{\Xi}}$

Result: Real-valued objective function metric, d

begin

  for $1 \leq j \leq J$ **do**

    Set $\hat{\mathbf{s}}_i, j$ to sensor optimization based on $\hat{\boldsymbol{\Xi}}, \mathbf{m}_j, \mathbf{a}_j, \mathbf{b}_{i,j,k}$.

  end

  Set $d = 0$ **for** $1 \leq j_1 \leq J$ **do**

    for $j_1 \leq j_2 \leq J$ **do**

      Set $d = d + \|\hat{\mathbf{s}}_{i,j_1} - \hat{\mathbf{s}}_{i,j_2}\|$

    end

  end

  Return d

end

To illustrate how this method is expected to work, a diagram is depicted in Figure

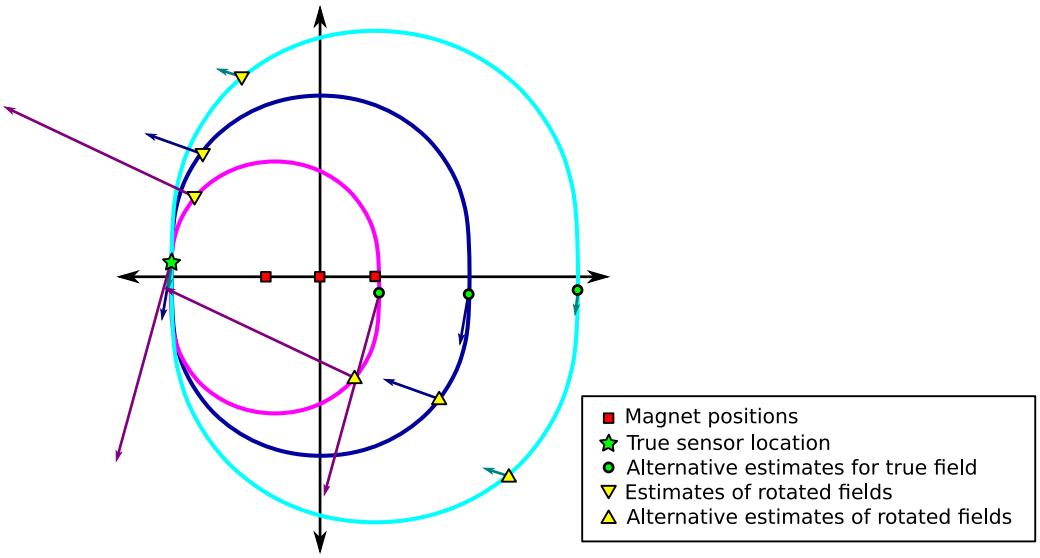


Figure 48. Illustration of the sensor estimate alignment method reduced to a 2-D plane. The curves represent the set of values with a constant B-field strength as caused by each of the three magnets. The arrows represent the B-field measured at the various labeled points where the arrow lengths are proportional to the field strengths.

48. The problem is depicted where the magnets and the sensor all lie on a common plane with the magnetic moment of each magnet pointed upward. The curves represent positions of constant B-field amplitudes. The true magnetic sensor position is located at the position of the star. At the true sensor position, the measured B-fields give estimates that are all equivalent. Due to symmetry, each of the magnetic sensor measurements can represent measurements from two possible locations. Selecting which of the estimates is correct can be done either by using outside information (such as all sensors are above the plane $z = 0$) or the set of solutions that minimizes the distance metric can be selected. Here the alternate solutions for the accurate fields are depicted by circles. As the magnetic fields are rotated, a different set of estimated solutions are found. Here the vectors were rotated by 100° and the new estimated locations were found (depicted by down arrows) and the associated alternative estimates (depicted by up arrows).

10.4 Quick Estimation of Sensor Positions

The biggest problem with performing this optimization is that within each evaluation of the objective function, there must be the execution of a separate sensor estimate for *each* magnet position. This optimization must be performed over a set of sample values. When performing an optimization without a Jacobian, this objective function is too time-consuming to be practical.

There is a way around this, however, by utilizing the fast spherical estimate described in Section 8.4.

The first step is to perform a sensor position estimate where $\boldsymbol{\Xi} = \mathbf{I}$. This function is calculated once for each j to provide an initial sensor estimate $\tilde{\mathbf{s}}_{i,j}$. Once this value has been computed, it is used to identify a representative B-field vector, $\tilde{\mathbf{b}}_{i,j} = \mathbf{b}(\tilde{\mathbf{s}}_{i,j}, \mathbf{a}_j, \mathbf{m}_j)$.

Now, instead of performing a full optimization procedure, the objective function performs a rotation of $\tilde{\mathbf{b}}_{i,j}$ and it uses the fast estimate algorithm to find new sensor estimates. This reduces the problem from an optimization of three parameters over K samples that takes on the order of seconds to compute to an operation that optimizes one parameter over a single sample that takes on the order of milliseconds to compute. Thus, although the initial estimate of the sensor positions can be time consuming with an execution that increases at best linearly with number of samples, the execution after the initial sensor position optimizations has an execution with complexity $\mathcal{O}(1)$.

When testing this algorithm, it was discovered that there is apparently a local minimum in the objective function, indicating that the problem is not convex. It can be quite difficult to visualize where the local minima exist and it is a function of the testing magnet position and orientation values. The quantities of the evaluation of the objective function at the false estimates that represent local minima are much larger values than the objective function evaluated at the correct orientation.

To address this possibility of finding a local minimum that is not the global minimum, the process was implemented multiple times: once for each sensor. Each of these functions provide a different estimate of the rotation, herein denoted as $\boldsymbol{\Xi}_l$ where l corresponds to the index of the sensor number that provided the estimate. The index here is denoted by l instead of i to distinguish it from a sensor index in a later optimization step. This added computation increases the execution to a complexity $\mathcal{O}(J)$, though the calculation is efficient enough that the initial optimization of sensor positions still dominates the execution time.

An alternative method can be performed by starting with different initial conditions. This can be done deterministically by composing the rotation estimate $\boldsymbol{\Xi}$ with another rotation to maximize distance of initial conditions and running another optimization, or it could be done stochastically by finding a random rotation and then performing an optimization.

To specify the final orientation estimate $\hat{\boldsymbol{\Xi}}$, the objective functions were evaluated for each sensor using each of the different estimates and the objective function that provided the lowest L^2 amplitude is chosen as the best fit.

$$\hat{\boldsymbol{\Xi}} = \arg\min_{\boldsymbol{\Xi}_l} \sum_i f^2_{\text{SEAM2}}(i, \tilde{\mathbf{b}}_{i,j}, \mathbf{a}_j, \mathbf{m}_j, \boldsymbol{\Xi}_l) \tag{205}$$

Once a final value of $\boldsymbol{\Xi}$ has been selected, the estimates for the sensor positions can be calculated (or be kept from the latest evaluation of the objective function. Alternatively, a full optimization can be run using the new rotation estimate. In practice, this is unnecessary. The initial optimization seems to be accurate up to some quantization error from the rotation.

An overview of the algorithm is outlined in Algorithm 2.

Data: $\mathbf{b}_{i,j,k}, \mathbf{a}_j, \mathbf{m}_j$

Result: Estimate of $\boldsymbol{\Xi}, \mathbf{s}_i$

begin

  for *each sensor* i **do**

    for *each magnet* j **do**

      Using $\boldsymbol{\Xi} = \mathbf{I}$, estimate sensor position $\tilde{\mathbf{s}}_{i,j}$

      Set $\tilde{\mathbf{b}}_{i,j} = \mathbf{b}(\tilde{\mathbf{s}}_{i,j}, \mathbf{a}_j, \mathbf{m}_j)$

    end

    Set $\boldsymbol{\Xi}_l$ to $\arg\min_{\boldsymbol{\Xi}} f_{\text{SEAM}}\left(i, \tilde{\boldsymbol{b}}_{i,j}, \mathbf{a}_j, \mathbf{m}_j, \boldsymbol{\Xi}\right)$

  end

  Set $\boldsymbol{\sigma}$ to $\mathbf{0}_{I\times 1}$

  for *each* $\boldsymbol{\Xi}_l$ **do**

    for *each sensor* i **do**

      $\sigma_l = \sigma_l + f^2_{\text{SEAM}}\left(i, \tilde{\boldsymbol{b}}_{i,j}, \mathbf{a}_j, \mathbf{m}_j, \boldsymbol{\Xi}_{l=i}\right)$

    end

  end

  Return the $\boldsymbol{\Xi}_l$ corresponding to $\arg\min_l \boldsymbol{\sigma}_l$ and corresponding $\mathbf{s}_i$.

end

Algorithm 2: Outline of the algorithm to estimate the rotation and the sensor positions simultaneously.

To visualize the optimization space, slices of the sample space is depicted in Figure 49. The space of (ψ, θ, ϕ) is a double covering of the space of rotations and the symmetry is apparent in the slices. The space is log-compressed to accurately show the contours. There is a natural saddle point at the region $\psi = 0$, which is constant for all vectors of rotation. Performing an optimization with the identity (i.e. no rotation $(0, \theta,$

phi)) can lead to slow optimizations. Consequently, the optimization is initialized with a random rotation.

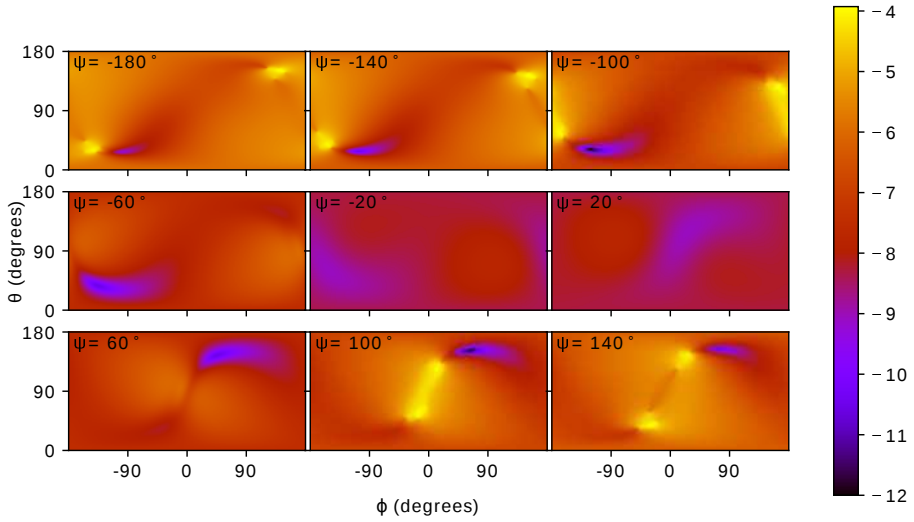


Figure 49. Log of objective function measurements for the objective function seeking to find the rotation Ξ. The horizontal axis represents the azimuth angle ϕ and the vertical axis represents the elevation angle θ. The slices represent different amounts of rotation ψ. The logarithm is taken to emphasize the location of the minima and the lower limit of the color bar is fixed since the minimum values are zero, and thus are singularities.

10.5 Evaluation

To begin, a description of the testing configuration is described. The system environment is set in a constant configuration. The earlier chapters describe considerations of optimization for the sensor positions and the orientations. For the technique used here, the sensor position algorithm is used internally and the orientation is optimized by a new metric based on the optimal sensor position estimates.

The sensor positions are located at the same locations in the prior chapters: $\mathbf{s}_1 = (1, -5.5, 1)$ cm, $\mathbf{s}_2 = (1, 5.5, 1)$ cm, $\mathbf{s}_3 = (-1, -7, 2)$ cm, and $\mathbf{s}_4 = (-1, 7, 2)$ cm. The magnet positions are at the origin and the four positions located 2cm from the center in the positive and negative x and y directions. All magnet positions are at $z = 0$. The magnetic moment points in the $+z$ direction. The code used for the sensor position estimators, however, is the same code used for the code that optimizes over the sensor position exclusively, so the offsets and the magnetic moment directions can

be easily changed, but it is assumed that the true values are known. The magnetic moment amplitude is the same as before ($14.80\ {}^{\mathrm{mA}}/_{\mathrm{m}^2}$).

The algorithm begins by doing a sensor position estimation with no rotation for each sensor and for each magnet configuration. In this case, that is $4 \times 5 = 20$ optimizations. The method used here can be either the Euclidean method or the spherical method as described in Chapter 8. From the results in that chapter, the accuracy between these two methods is quite similar and the Euclidean method is faster, so the results presented here use the Euclidean version. Both methods were successful, however.

After the initial sensor estimation, the model information is used to create a representative B-field vector for each sensor and magnet position configuration. From this point, the sensor estimation is done using the fast estimation using the representative field estimates and performing rotations on that. An independent optimization is run for each sensor to provide a candidate rotation using measurements taken from each magnet configuration. This provides multiple candidate rotations.

Experimental results have shown that there are some local minima that are not the global minima. The nature and location of these minima is a function of sensor positions and magnet positions and has not been fully characterized. There is an increased risk of having the different methods fall into localized minima if the initial conditions of the rotation is taken to be the identity, i.e. no rotation. To avoid this, each of the sensor optimizations uses a random initial rotation. With four sensors, usually three or four of them provide some expression of the correct rotation.

The proposed rotations from each sensor is then used to calculate the objective function for the other sensors and an L^2 metric of the vector of objective function results is used to evaluate. The candidate rotation that returns the lowest value is chosen as the true rotation.

At this point, it is possible to do another full optimization to try and find the

sensor positions conditioned on this new rotation candidate or it could be calculated using the fast approximation method. In practice, the results of the two versions are nearly identical up to quantization error from the rotation operation. Because each rotation configuration provides a different (hopefully nearly identical) estimate of the sensor position, and due to the low cost of a single fast estimation operation, the fast method is used for each magnet configuration and the results are averaged to give a final sensor position. To save some time, it could just use one magnet configuration for the estimate, but the extra execution time is negligible compared to the optimization.

Finally, the rotation estimate and the sensor positions are returned. These values are then compared using the established rotation metric and Euclidean distance between the true values and the estimates. These results were found for sample counts of $\{10,\ 30,\ 100,\ 300,\ 1000\}$. Gaussian noise is provided the same way as before with $\sigma = \{0,\ 0.000316,\ 0.001,\ 0.00316,\ 0.01,\ 0.0316,\ 0.1\}$ scaled by a multiplier of 0.4315 so the strength of the noise is approximately proportional to the strength of the signal. The set of parameters for this method is more constrained than in prior chapters to be closer to the range of values expected in a practical application, though the results from prior chapters can be used to infer the behavior for larger numbers of samples and under noisier conditions.

10.5.1 Execution Time

The execution time is compared under different noise conditions in Figure 50. Note that, unlike previous execution time plots, this figure has a logarithmic independent axis with a linear dependent axis. It can be seen that the runtime is essentially linear for small sample counts and then suddenly increases from 100 samples to 1000 samples. The reason for this is that the number of samples only impacts the initial optimization for finding the sensor positions under no rotation and generating the

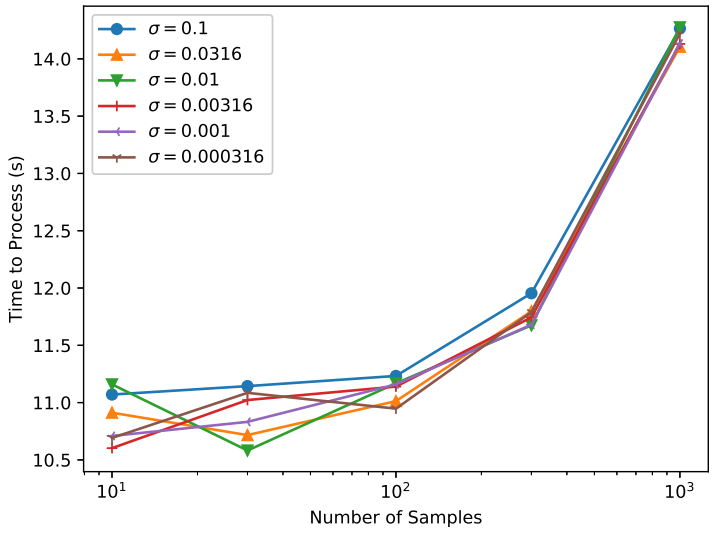


Figure 50: The runtime based on the amount of noise in the system.

representative B-field values. For small sample counts, this optimization is trivial compared to the more complex meta optimization, which has a complexity order $\mathcal{O}(1)$. At a certain point, this initial optimization becomes the more complex portion and the meta optimization becomes negligible. Thus the execution time can be extended by looking at the time required for performing sensor estimation alone.

The total time to perform the optimization in the existing Python framework is at least 10s and goes up to 15s for 1000 samples. It is also interesting to note that the noise has little discernible effect on the optimization time, though the more detailed analysis from Chapter 8 shows that there is some effect that becomes more pronounced as the noise levels reach a strength approximately equal to the signal strength (a level of noise not present in this analysis) when the initial sensor estimator dominates the execution time (i.e. above 100 samples.)

The fast estimate method takes between 2ms and 5ms to calculate. Comparing to a full analysis which can take between 5-10ms for small sample counts to 1-4 sec for 10^4 samples, this simplification is essential for efficient optimization.

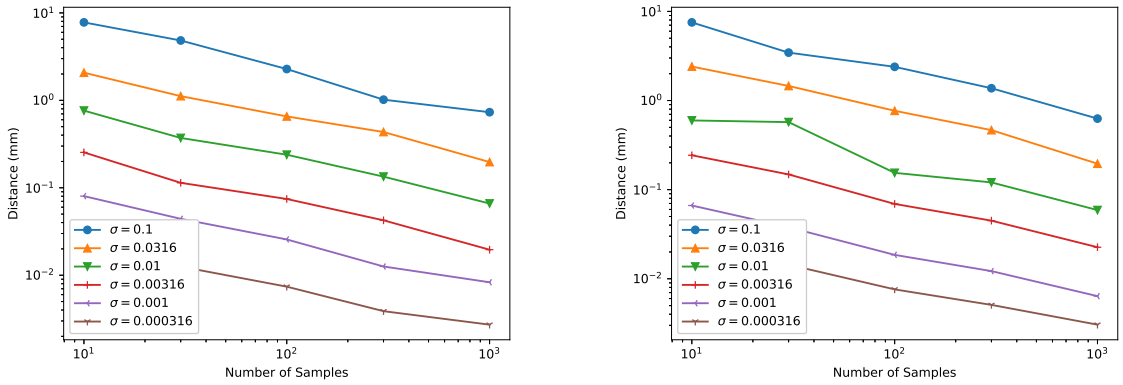
10.5.2 Sensor Position Accuracy

The accuracy is analyzed as a function of the noise with increasing numbers of samples. These results are in Figure 51. The top two figures represent the sensors which are closer to the headset. The error is measured in mm. With a noise approximately equal to the signal strength and with only 10 samples, the error is approximately 1cm, but with 1000 samples, this can be reduced to within 1-2mm. In practice, the noise level should ideally be between 1-3% to be able to be fairly accurate. This is about 3 times more error than in the case where the rotation is known exactly and the magnetic moments were randomly generated. Because the vectors from the magnet to the sensors is close to perpendicular to the magnetic moment, it is expected that there is some additional error due to the symmetry issue described in Chapter 8 as well as from the error in the estimate of the rotation angle. Placing the magnet with a magnetic moment pointing towards the sensors could lead to some improvement in the estimates, though the cylindrical magnet being placed flat on a table is more natural in practice following the methods of the experiment.

A comparison of the error between the true position and the estimate when the true rotation is known compared to the current method where the rotation is being estimated simultaneously is presented in Figure 52 for sensor 1. The plot on the left uses a Euclidean estimate method. The error for the rotation and position estimator is between two and three times the error compared to the method where the rotation is known exactly. Even in this case, though, it is quite possible to get the error within 1mm under expected operating conditions.

10.5.3 Rotational Accuracy

The accuracy estimate as a function of the noise for a variety of sample counts is presented in Figure 53.



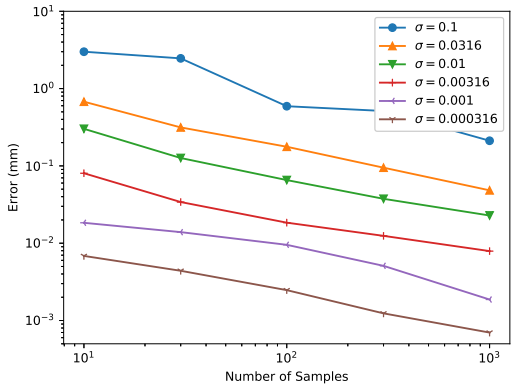
(a) Sensor 1

(b) Sensor 2

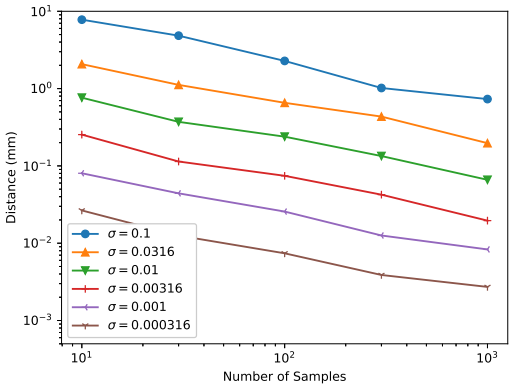
(c) Sensor 3

(d) Sensor 4

Figure 51. Comparison of the accuracy of the sensor estimates (using a Euclidean distance) for the four different sensors. The first two sensors (top) are slightly closer to the magnet positions than the second two sensors (bottom). The accuracy is parameterized by the noise levels with standard deviations as specified in the legends. The noise standard deviations are multiplied by a constant 0.4315 G to normalize to an amplitude approximately proportional to the signal strength.



(a) Sensor 1 error when the rotation is known



(b) Sensor 1 error when rotation is estimated

Figure 52. The plot on the left shows the error of the estimate when the underlying rotation is known explicitly. The plot on the right shows the error of the sensor position estimates in the method when the rotation is being estimated simultaneously.

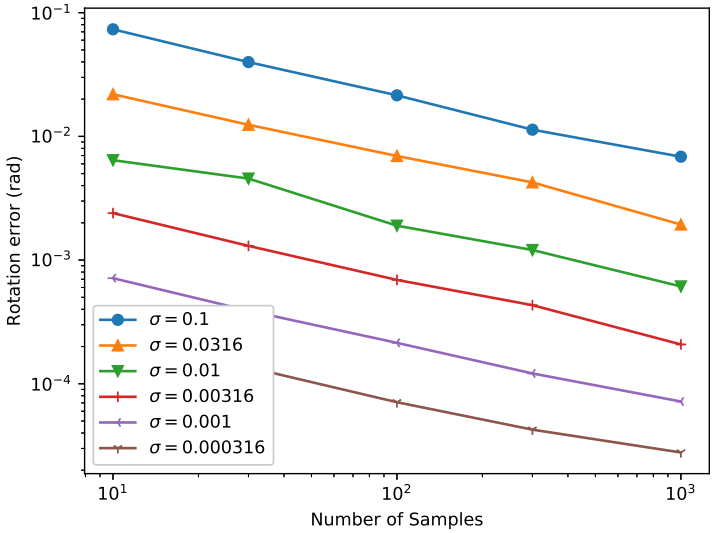
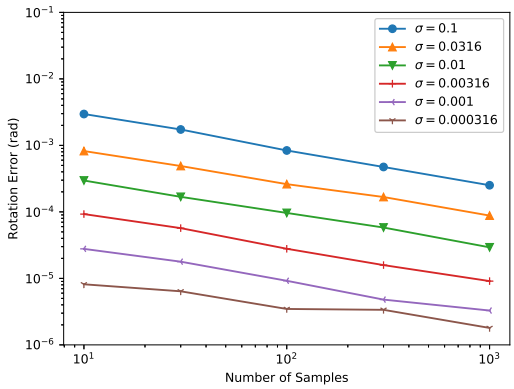
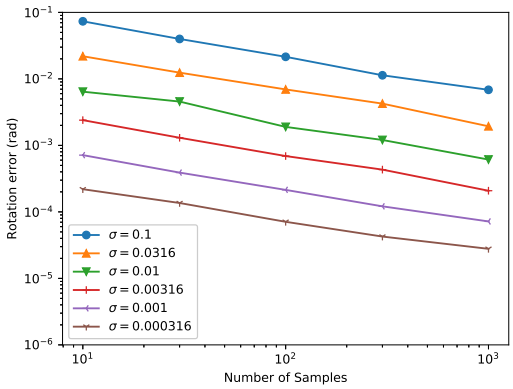


Figure 53. The amount of rotational error in radians according to the rotational metric as a function of the amount of noise.



(a) Rotation optimization when sensor positions are known



(b) Rotation and sensor position optimization

Figure 54. The plot on the left is the rotational error for a system using both magnet values and sensor position information with a Euclidean optimization with a known set of sensor positions. The plot on the right is the rotational error found when the sensor positions are being estimated along with the rotation. The noise parameters are equivalent between the two experiments with the same underlying sensor and magnet positions with noise being scaled by a common multiplier of 0.4315 G.

These results are compared with the system that used both magnet position information and sensor position information when the sensor positions are known explicitly. These are pictured side-by-side in Figure 54. Clearly having an exact value of the sensor positions and using all the data for rotation is better, the rotation estimate is still likely acceptable when both rotation and sensor position are being calculated simultaneously.

CHAPTER XI

CONCLUSION AND FUTURE WORK

In developing a system to do proportional control, it was discovered that the ability to properly perform tracking in such a system requires good calibration and configuration steps. Various aspects of this problem were addressed in this work with the intention of providing a framework suitable for use with existing tracking methods. This chapter briefly outlines the conclusions from the research, the contributions of the work, and a list of future derivative work.

11.1 Conclusions

From the work outlined, it is possible to, using sets of data collected through magnetic sensor measurements, to perform calibration for the headset. The accuracy of this calibration is impacted somewhat by magnetic fields present during the experiment. Research on this has been done in the past, but the steps used for the proposed method are outlined explicitly and some analysis of these estimators was performed.

To be able to do proper data fusion, it is necessary to align the different magnetic sensors. A few methods to accomplish this were proposed and the results of the effectiveness were presented. This can allow for the sensor information to be fused together for later configuration steps. The presented methods are based on existing work in trying to align magnetic sensors and accelerometers, for instance, as used in robotics. The work presented shows how this can be accomplished using exclusively the magnetic sensor information.

With the data fusion complete, it is necessary to find the orientation to align the system with the physical world. To do this, a method of collecting data in the presence of a known magnet was presented. Under the conditions of knowing the

magnetic moment information and the magnetic position, it was demonstrated that, given a rotation to the correct world reference orientation, it is possible to find the sensor position from data. To accomplish correct optimization of this non-convex problem, a method for finding a good initial estimate was provided. Two different algorithms were also presented to accomplish sensor position estimation.

Given a known set of sensor positions, some methods were presented for estimating a rotation to align the sensor data to the true world orientation reference frame. Three different techniques were used for this and the results were presented.

Finally, a technique for simultaneously finding the global rotation and the sensor estimates was presented. While the rotation and the sensor estimates were less accurate than the counterparts under known parameters, it is still possible under proper noise conditions and with a sufficient number of samples, to get a good estimate of the headset configuration.

These steps provide a framework for calibrating and configuring the headset as described in preparation for doing a variety of target-tracking methods such as non-linear Kalman filtering or particle filter tracking.

11.2 Contributions

As part of an SBIR, independent of the work of this thesis, I created firmware and helped to develop the hardware to implement a headset. Although this is not directly a part of the effort for this thesis, the experience in developing the hardware and the implementation of existing algorithms has demonstrated the ability to create a physical framework that is capable of moving towards more sophisticated tracking methods. This work also demonstrated the need of properly configuring the system for tracking methods to be successful.

The work done on calibration is based on existing literature, but some new analysis was done as well as a framework to assess the accuracy of this optimization. The

results presented here explicitly take noise into account and provide a resource to identify what level of accuracy should be expected under different noise conditions. Additionally, different path methods were taken into account and some novel analysis of these methods was provided. A new method for making an estimate of a reference alignment with two pairs of rotation vectors was also presented. This technique is sufficient in a noise-free case when two different vector pairs are provided, but it is also useful in a noisy case to make an initial guess and speed up rotational estimations.

The work in orienting the different magnetic sensors to a common orientation is inspired by existing work. The novelty of this effort is that it does not depend on any additional information from accelerometers or gyroscopes, since the magnetic measurements are the important element. The work presented demonstrates the expected accuracy under a variety of sample sizes and noise conditions. This work also takes into account the different paths that could be used in the training process and shows how the paths impact the results. A quick estimator for finding the sensor position was presented based on spherical vector quantities. This can be used individually for efficient estimations or can be used to find an initial guess for a more complicated system.

The sensor estimation and rotation estimation are new to this work. This research shows, both mathematically and through simulated results, the limitations of the methods as well as how the relative position of the magnet to the sensors when compared to the magnetic moment impact the accuracy of these techniques. This can be used to help inform the best placement of magnetic sensors on a headset. While an initial EM-type algorithm was unsuccessful in accurately estimating sensor position and a rotational alignment to the real-world reference frame, the novel technique presented shows that it is possible to create an algorithm to accomplish this task. To make this problem efficient, the quick estimator was used. With this tool available, the solution can be found in tens of seconds under most conditions.

11.3 Future Work

There are many areas of future work available. The first one is to implement all the simulated methods to the physical system and analyze the results. There have been some limitations to accessibility to the hardware that have prevented this from being done already, but the code designed under this thesis was done in a way that is compatible with the existing hardware being used by Bionic Sciences, Inc.

Once the framework presented herein has been fully tested in a physical system, the next step is to use nonlinear Kalman filtering and particle filtering to track a magnet. The work presented here is useful to assuring that this effort can be successful. This is especially true for the considerations of the sensor positioning in relation to the magnetic moment.

Because the sensors on the headset are not static, it is important to be able to accurately adjust the model based on these changes. While this should not effect the first stage of the system, this does impact all other stages. Prior work has been done to, using accelerometers and gyroscopes, recognize when an adjustment has been made. There is still work to do to adjust the rotation alignment of the sensors and update the sensor positions in the global reference frame. The first step would be to recognize when the headset is no longer trained, and then find a system that is capable of dynamically adjusting to these changes. It is still somewhat uncertain what level of success could be expected in these cases, but it seems that it could be possible to create particle filters based on the techniques described in this thesis to address these challenges.

Other future work would focus on how to best use the positional information acquired through these techniques to create a set of commands to be issued by the headset. There will be some amount of error in the system, but future work will focus on, within this uncertain framework, finding techniques for generating useful commands from the tracking data.

REFERENCES

[1] "Ability World." http://www.ability-world.com. Accessed: 2014-10-13.

[2] "Arrington Research." http://www.arringtonresearch.com. Accessed: 2014-10-13.

[3] "Blackrock Neuromed." http://www.blackrockneuromed.com. Accessed: 2014-10-13.

[4] "Cephalon." http://cephalon.eu. Accessed: 2014-10-13.

[5] "IntegraMouse Plus." http://integramouse.com. Accessed: 2014-10-11.

[6] "Magnetic field calculators." https://www.ngdc.noaa.gov/geomag-web. Accessed:2018-05-24.

[7] "ATWiki." http://atwiki.assistvetech.net/index.php/Alternative_wheelchair_control, 2010. Accessed: 2014-10-11.

[8] Adafruit, I., "Bluefruit nRF52 Feather pinout." https://learn.adafruit.com/assets/43921, 2018. Accessed:2018-09-27.

[9] Babcock, J. S. and Pelz, J. B., "Building a lightweight eyetracking headgear," in *Proceedings of the 2004 symposium on Eye tracking research & applications*, pp. 109–114, ACM, 2004.

[10] Ball, L., Nordness, A., Fager, S., Kersch, K., Pattee, G., and Beukelman, D., "Eye-gaze access of AAC technology for persons with amyotrophic lateral sclerosis," *Journal Medical Speech Language Pathology*, vol. 18, pp. 11–23, 2010.

[11] Bilmes, J. A., Li, X., Malkin, J., Kilanski, K., Wright, R., Kirchhoff, K., Subramanya, A., Harada, S., Landay, J. A., Dowden, P., and others, "The vocal joystick: A voice-based human-computer interface for individuals with motor impairments," in *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pp. 995–1002, Association for Computational Linguistics, 2005.

[12] Bonnet, S., Bassompierre, C., Godin, C., Lesecq, S., and Barraud, A., "Calibration methods for inertial and magnetic sensors," *Sensors and Actuators A: Physical*, vol. 156, no. 2, pp. 302–311, 2009.

[13] Bulling, A., Roggen, D., and Trster, G., "It's in your eyes: towards context-awareness and mobile HCI using wearable EOG goggles," in *Proceedings of the 10th international conference on Ubiquitous computing*, pp. 84–93, ACM, 2008.

[14] Carpenter, J., Clifford, P., and Fearnhead, P., "Improved particle filter for nonlinear problems," *IEE Proceedings-Radar, Sonar and Navigation*, vol. 146, no. 1, pp. 2–7, 1999.

[15] Chulliat, A., Macmillan, S., Alken, P., Beggan, C., Nair, M., Hamilton, B., Woods, A., Ridley, V., Maus, S., and Thomson, A., "The us/uk world magnetic model for 2015-2020," 2015.

[16] Cook, A. M. and Polgar, J. M., *Cook and Hussey's assistive technologies: principles and practice*. Elsevier Health Sciences, 2013.

[17] Cullity, B. D. and Graham, C. D., *Introduction to magnetic materials*. John Wiley & Sons, 2011.

[18] Davis, J., "Mathematical modeling of earths magnetic field," *Technical Note, Virginia Tech, Blacksburg*, 2004.

[19] Decker, B. L., "World geodetic system 1984," tech. rep., Defense Mapping Agency Aerospace Center St Louis Afs Mo, 1986.

[20] DEVELOPERS, S., "Scipy optimization." https://docs.scipy.org, 2018. Accessed:2018-05-31.

[21] EDELSTEIN, A. S. and FISCHER, G. A., "Minimizing 1/f noise in magnetic sensors using a microelectromechanical system flux concentrator," *Journal of applied physics*, vol. 91, no. 10, pp. 7795–7797, 2002.

[22] ENDRE, S. and MAYERS, D., "An introduction to numerical analysis," *Cambridge, UK*, 2003.

[23] FARAJIDAVAR, A., BLOCK, J. M., and GHOVANLOO, M., "A comprehensive method for magnetic sensor calibration: A precise system for 3-d tracking of the tongue movements," in *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE*, pp. 1153–1156, IEEE, 2012.

[24] GEORGE JR, A. and OTHERS, *Essentials of Padé approximants*. Elsevier, 1975.

[25] GHOVANLOO, M., "Tongue operated assistive technologies," *IEEE engineering in medicine and biology magazine*, vol. 1, p. 4376, 2007.

[26] GOLAN, J. S., *The linear algebra a beginning graduate student ought to know*. Springer Science & Business Media, 2012.

[27] GOLUB, G. H. and VAN LOAN, C. F., *Matrix computations*, vol. 3. JHU Press, 2012.

[28] GOOGLE, "Google elevation api." https://developers.google.com/maps/documentation/elevation, 2018. Accessed:2018-05-31.

[29] GOOGLE, "Google geocoding api." https://developers.google.com/maps/documentation/geocoding, 2018. Accessed:2018-05-31.

[30] HALL, B., *Lie groups, Lie algebras, and representations: an elementary introduction*, vol. 222. Springer, 2015.

[31] HORN, B. K., HILDEN, H. M., and NEGAHDARIPOUR, S., "Closed-form solution of absolute orientation using orthonormal matrices," *JOSA A*, vol. 5, no. 7, pp. 1127–1135, 1988.

[32] HUANG, C.-N., CHEN, C.-H., and CHUNG, H.-Y., "Application of facial electromyography in computer mouse access for people with disabilities," *Disability & Rehabilitation*, vol. 28, no. 4, pp. 231–237, 2006.

[33] HUYNH, D. Q., "Metrics for 3d rotations: Comparison and analysis," *Journal of Mathematical Imaging and Vision*, vol. 35, no. 2, pp. 155–164, 2009.

[34] INC., R., "BMD-340 module for Bluetooth 5 and 802.15.4." http://go.rigado.com/BMD-340-Data-Sheet, 2018. Accessed:2018-09-27.

[35] JACKSON, A., JONKERS, A. R., and WALKER, M. R., "Four centuries of geomagnetic secular variation from historical records," *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 358, no. 1768, pp. 957–990, 2000.

[36] JONES, M., GROGG, K., ANSCHUTZ, J., and FIERMAN, R., "A sip-and-puff wireless remote control for the Apple iPod," *Assistive Technology*, vol. 20, no. 2, pp. 107–110, 2008.

[37] K and J MAGNETICS, I., "Disc and cylinder neodymium magnets product list." https://kjmagnetics.com, 2018. Accessed:2018-09-18.

[38] KIM, S.-P., SIMERAL, J. D., HOCHBERG, L. R., DONOGHUE, J. P., FRIEHS, G. M., and BLACK, M. J., "Point-and-click cursor control with an intracortical neural interface system by humans with tetraplegia," *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 19, no. 2, pp. 193–203, 2011.

[39] KRANTZ, S. G., *Handbook of complex variables*. Springer Science & Business Media, 2012.

[40] LENZ, J. and EDELSTEIN, S., "Magnetic sensors and their applications," *IEEE Sensors journal*, vol. 6, no. 3, pp. 631–649, 2006.

[41] Leuthardt, E. C., Gaona, C., Sharma, M., Szrama, N., Roland, J., Freudenberg, Z., Solis, J., Breshears, J., and Schalk, G., "Using the electrocorticographic speech network to control a braincomputer interface in humans," *Journal of neural engineering*, vol. 8, no. 3, p. 036004, 2011.

[42] Moon, T. K. and Stirling, W. C., *Mathematical methods and algorithms for signal processing*, vol. 1. Prentice hall New York, 2000.

[43] Nelder, J. A. and Mead, R., "A simplex method for function minimization," *The computer journal*, vol. 7, no. 4, pp. 308–313, 1965.

[44] Nichols, E., Morrison, H., and Clarke, J., "Signals and noise in measurements of low-frequency geomagnetic fields," *Journal of Geophysical Research: Solid Earth*, vol. 93, no. B11, pp. 13743–13754, 1988.

[45] Nordic Semiconductor, I., "nRF52832 product specification v1.4." `https://http://infocenter.nordicsemi.com/pdf/nRF52832_PS_v1.4.pdf`, 2017. Accessed:2018-09-27.

[46] Novak, D., Mihelj, M., and Munih, M., "Dual-task performance in multimodal human-computer interaction: a psychophysiological perspective," *Multimedia Tools and Applications*, vol. 56, no. 3, pp. 553–567, 2012.

[47] Nutt, W., Arlanch, C., Nigg, S., and Staufert, G., "Tongue-mouse for quadriplegics," *Journal of Micromechanics and Microengineering*, vol. 8, no. 2, p. 155, 1998.

[48] Phillips, B. and Zhao, H., "Predictors of assistive technology abandonment," *Assistive Technology*, vol. 5, no. 1, pp. 36–45, 1993.

[49] Popovic, R. S., *Hall effect devices*. CRC Press, 2003.

[50] Power, S. D., Kushki, A., and Chau, T., "Towards a system-paced near-infrared spectroscopy braincomputer interface: differentiating prefrontal activity due to mental arithmetic and mental singing from the no-control state," *Journal of neural engineering*, vol. 8, no. 6, p. 066004, 2011.

[51] Quinn, J., "Geomag driver."

[52] Rui, Y. and Chen, Y., "Better proposal distributions: Object tracking using unscented particle filter," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 2, pp. II–II, IEEE, 2001.

[53] Sabatini, A. M., "Quaternion-based extended kalman filter for determining orientation by inertial and magnetic sensing," *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 7, pp. 1346–1356, 2006.

[54] Salem, C. and Zhai, S., "An isometric tongue pointing device," in *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*, pp. 538–539, ACM, 1997.

[55] Saponas, T. S., Kelly, D., Parviz, B. A., and Tan, D. S., "Optically sensing tongue gestures for computer input," in *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, pp. 177–180, ACM, 2009.

[56] Shanno, D. F., "Conditioning of quasi-newton methods for function minimization," *Mathematics of computation*, vol. 24, no. 111, pp. 647–656, 1970.

[57] Sibert, L. E. and Jacob, R. J., "Evaluation of eye gaze interaction," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 281–288, ACM, 2000.

[58] Simeral, J., Kim, S., Black, M., Donoghue, J., and Hochberg, L., "Neural control of cursor trajectory and click by a human with tetraplegia 1000 days after implant of an intracortical microelectrode array," *Journal of neural engineering*, vol. 8, no. 2, p. 025027, 2011.

[59] STMicroelectronics, *Ulta-compact high-performance eCompass module: 3D accelerometer and 3D magnetometer*, Nov 2013. Rev. 2.

[60] STMicroelectronics, "Ultra-compact high-performance eCompass module: 3D accelerometer and 3D magnetometer." `https://www.st.com/resource/en/datasheet/lsm303d.pdf`, 2013. Accessed:2018-09-27.

[61] STMicroelectronics, "iNEMO inertial module: 3D accelerometer, 3D gyroscope, 3D megnetometer." `https://www.st.com/resource/en/datasheet/lsm9ds1.pdf`, 2015. Accessed:2018-09-27.

[62] STMicroelectronics, *iNEMO inertial module: 3D accelerometer, 3D gyroscope, 3D magnetometer*, March 2015. Rev. 3.

[63] Struijk, L. N. A., "An inductive tongue computer interface for control of computers and assistive devices.," *IEEE Trans. Biomed. Engineering*, vol. 53, no. 12, pp. 2594–2597, 2006.

[64] Thébault, E., Finlay, C. C., Beggan, C. D., Alken, P., Aubert, J., Barrois, O., Bertrand, F., Bondar, T., Boness, A., Brocco, L., and others, "International geomagnetic reference field: the 12th generation," *Earth, Planets and Space*, vol. 67, no. 1, p. 79, 2015.

[65] Usakli, A. B., Gurkan, S., Aloise, F., Vecchiato, G., and Babiloni, F., "On the use of electrooculogram for efficient human computer interfaces," *Computational intelligence and neuroscience*, vol. 2010, p. 1, 2010.

[66] Vaidyanathan, R., Chung, B., Gupta, L., Kook, H., Kota, S., and West, J. D., "Tongue-movement communication and control concept for hands-free humanmachine interfaces," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 37, no. 4, pp. 533–546, 2007.

[67] Van Der Merwe, R., Doucet, A., De Freitas, N., and Wan, E. A., "The unscented particle filter," in *Advances in neural information processing systems*, pp. 584–590, 2001.

[68] Vince, J., *Quaternions for computer graphics*. Springer Science & Business Media, 2011.

[69] Volosyak, I., "SSVEP-based bremenBCI interfaceboosting information transfer rates," *Journal of neural engineering*, vol. 8, no. 3, p. 036020, 2011.

[70] Wan, E. A. and Van Der Merwe, R., "The unscented Kalman filter for nonlinear estimation," in *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, pp. 153–158, IEEE, 2000.

[71] Wan, E. A. and Van Der Merwe, R., "The unscented kalman filter," *Kalman filtering and neural networks*, pp. 221–280, 2001.

[72] Wolpaw, J. R., Birbaumer, N., McFarland, D. J., Pfurtscheller, G., and Vaughan, T. M., "Braincomputer interfaces for communication and control," *Clinical neurophysiology*, vol. 113, no. 6, pp. 767–791, 2002.